School of Computation, Information and Technology

Technische Universität München

Master´s Thesis in Computational Science and Engineering

# Full Waveform Inversion with Neural Network Based Ansatzfunctions

Rodrigo Sánchez Cela

# School of Computation, Information and Technology

Technische Universität München

Master's Thesis in Computational Science and Engineering

# Full Waveform Inversion with Neural Network Based Ansatzfunctions

| | |
|---|---|
| Author: | Rodrigo Sánchez Cela |
| Examiner: | Dr. Felix Dietrich |
| Assistant advisor: | Qing Sun |
| Submission Date: | September 14th, 2024 |

I hereby declare that this thesis is entirely the result of my own work except where otherwise indicated. I have only used the resources given in the list of references.

September 14th, 2024

Rodrigo Sánchez Cela

# Acknowledgements

As I write the last sentences of my final assignment in the MSc. Computational Science and Engineering of the Technische Universität München, I want to express my gratitude to essential people in this new academic achievement. First and foremost, I want to say thank you to my supervisor, Qing Sun, for always being available and willing to help. I have heard countless times PhD students say the most critical factor when pursuing doctorate studies is your supervisor. I can confidently say I would be thrilled to pursue a PhD under your supervision if you were a Professor. Furthermore, I am thankful to Dr. Felix Dietrich. Thank you for proposing the topic of this master's thesis, allowing me to conduct this important academic milestone in your research group, and guiding the project when adversities arose. Moreover, I am incredibly grateful for the family I have - their unconditional love and support (as well as their financial support) have helped me be successful far away from home. Lastly, I want to make a special mention to my uncle Coco - who has been part of my decision to study abroad, has closely followed every step I have taken, and is a big part of the excellent cook I am right now, approved by my girlfriend. That's it, folks; it's time for new adventures.

# Abstract

Computer imaging techniques are crucial in fields like medicine and engineering, where CT scans help diagnose patients, and non-destructive testing (NDT) ensures the safety of structures such as airplane wings. In this thesis, we study the full waveform inversion (FWI) problem from a deep learning point of view. FWI, initially developed in seismology and applied recently in NDT, is explored using various neural network architectures, namely convolutional and feed-forward neural networks. Our initial goal was to apply a feed-forward neural network (FNN), along with the ”Sample Where It Matters” (SWIM) weight sampling algorithm, to solve the FWI problem outlined in [20]. We wanted to propose a new method for the research described in the cited article. The approach sought to reduce the number of trainable parameters and hence inference time by fixing hidden layer weights and only optimizing the output layer. However, our FNN failed to predict the objective material distribution, even after adjusting and testing out different initializations designed to make the algorithm converge easier, using different SWIM domain approximations. A supervised learning experiment confirmed the network could not approximate the discontinuous ground truth gamma using gradient descent, which explained the failure in solving the FWI problem. Seeking to determine the capabilities of neural networks to solve the FWI problem, we tested smoother ground truth functions: (i) a Gaussian and (ii) a sinusoidal function. In addition, we have also studied the effect of simulating a larger domain. Convolutional neural networks (CNNs) could solve both supervised learning and FWI tasks, independent of the discretization and domain size, whereas FNNs failed. With an extended domain and the same number of grid points as in the original experiments, an FNN with four hidden layers and 500 neurons per layer successfully handled smooth functions in the supervised learning framework, successfully solved the FWI problem for the sinusoidal-shaped function, however struggled with the Gaussian function in FWI. Lastly, we compared two CNN initializations for FWI on smooth functions: (i) Xavier-Glorot and (ii) weights trained on a voidless domain. We found that the first worked better for the Gaussian function, while the second was more effective for the sinusoidal function. In this thesis, we also discuss improvements and future work that could help answer our study’s unresolved questions.

# Contents

# 1. Introduction

Scientific computing is a branch of science with a fast evolution over the last decades. It is a multidisciplinary field at the intersection of applied mathematics, computer science, and engineering aiming at developing informative, problem-solving methodologies and robust algorithms for numerical simulations of physical systems. A simulation of a physical system can be, for example, the propagation of a sound or electromagnetic wave in some medium or modelling the pressure distribution of an aircraft while flying. Numerical simulations bring many advantages for scientists; for example, scientists can explore physical systems under a wide range of conditions by varying environment variables and studying the impact of these changes on the system. Moreover, some experiments might be too expensive, too dangerous, or simply impossible to undergo in a lab and a numerical model of the system allows us to overcome these barriers.

There are two broad categories of problems in which simulations of physical systems may fall into. As a first example, we may want to know how a fluid propagates in a given channel. For this purpose, the density of the fluid is known and we want to approximate Navier-Stokes equations to simulate how the fluid's pressure and velocity evolves in time and space. These types of problems, where we are certain about some causal factors and we want to model the effects of these causal factors, are referred to as *forward problems*. On the other hand, we may also want to find out information about the actual fluid, for example, its density. For this purpose, we need some observations, taken in an experimental setting, to infer its causes. These types of problems, where we need to infer the causes from the effects, are called *inverse problems*. Inverse problems are challenging to solve because generally an analytical solution does not exist. We need to recur to iterative methods to solve this type of problem. The gigantic progress in computing systems' capabilities as well as the advances in applied mathematics, in particular, numerical methods, have been essential for all the milestones achieved in recent years regarding the solution of inverse problems. Good examples of fields where an inverse problem constitutes the core of the field are medical imaging, and non destructive testing (NDT). The work in this master's thesis focuses on the same inverse problem tackled in NDT, and resolved using Full Waveform inversion (FWI).

FWI originated in seismology to retrieve accurate Earth subsurface models. In this master's thesis, we explore the FWI algorithm in the context of NDT with an additional twist: deep learning. Innovations and advances machine learning are beginning to impact scientific fields, such as the Earth sciences [23], becoming a hot topic in research in the last decade. AI is a trendy research field, but other classical scientific fields are seeing themselves re-invented with machine learning approaches. We are witnessing increasing research in using deep neural networks for solving partial differential equations (PDEs) as some classical methods encounter problems when dealing with significant non-linearities. The work presented in this master's thesis explores this direction, combining a classical approach to solve the scalar wave equation and novel neural network techniques to solve the inverse problem in FWI.

In this master's thesis we want to explore the FWI algorithm in combination with neural networks to model the density distribution of a material in the context of non-destructive testing. Initially, our aim was to design and implement a new method to solve the FWI problem researched in the article [20]. Results obtained after conducting the original experiments and implementing the new algorithm motivated us to look at a more general picture. With this master's thesis we want to answer the following research question: What are the capabilities of neural networks to solve a full waveform inversion problem? For this purpose, we design and conduct different experiments.

- Showcase the potential of a new proposed method to solve the FWI problem studied in article [20], and study the effect of different initializations in the final prediction.
- Explore FNNs and CNN capabilities for solving the FWI problem with smooth functions in two different domain sizes.
- Explore the effect of two initializations when solving the FWI challenge for smooth functions in an extended domain with CNN.

Previous research that explores similar scenarios can be found in chapter 2, which summarises and explains the state-of-the-art of this work's topic. The main work of this thesis is presented in chapter 3. Here, we provide a detailed explanation of our methodology, present the results of our experiments, and discuss our observations. Finally, in chapter 4, we summarise our findings, draw our conclusions, and outline potential future work.

# 2. State of the Art

## 2.1. Classical Full Waveform Inversion

Full Waveform Inversion (FWI) is an imaging method originating in seismology, aiming to improve the resolution of tomographic images from when it was developed to construct accurate subsurface models for a better understanding of subsurface properties. Patrick Lailly and Albert Tarantola introduced FWI in the early 1980s [38, 69]. This once novel method reveals information about the internal structure of a medium in a noninvasive manner and rapidly got recognition from the scientific community, extending to other scientific fields outside seismology; for example, in medical applications [18], and Non-Destructive Testing (NDT) [56], due to its capabilities for solving the inverse problem.

Irrespective of whether we consider a seismology problem, an NDT problem, or a medical imaging problem, FWI is a computational technique used to simulate the material distribution of a specimen. The FWI algorithm encompasses different steps as shown in figure 2.1. The essence of Full Waveform inversion is a model-based optimisation algorithm aiming at recovering the material distribution of a specimen. This method compares a simulated wavefield propagated in a numerical model of the material specimen to measurements taken in an experimental setting. The recorded measurements at the sensors carry information about the intrinsic properties of the material. The density, as well as reflections inside the material have a noticeable impact on the recorded measurements. The specimen's numerical model is iteratively updated so that the simulated wavefield approaches the experimental measurements in each iteration. Simulating a wavefield, comparing the misfit, and updating the specimen's model is repeated until the material distribution of the specimen is retrieved. Before running the optimization algorithm, it is imperative to record an initial set of measurements taken at specific points on the surface of the specimen as a reference for comparison. In addition, we start with an initial estimate of the specimen's material distribution and refine it iteratively. We strive to come closer to the actual material distribution with each update. Minimising the misfit between the set of wavefield measurements taken at collocation points on the specimen's surface and a simulation of the wavefield at the same collocation points with the current state of the specimen's model drives the convergence to the ground truth material distribution. This simulated wavefield results from numerically solving the wave equation with the current model parameters at every iteration. At each iteration, the algorithm refines the model's parameters such that the simulated wavefield approaches the recorded measurements. Generally, gradient-based optimisation methods are the preferred tool to update the model's parameters due to the high dimensionality and expensive misfit function [6]. Formally, FWI is a non-linear optimisation problem where the material parameters are the variables we seek to optimize. Formulating the problem as a least squares problem allows us to solve the non-linear optimisation task efficiently.

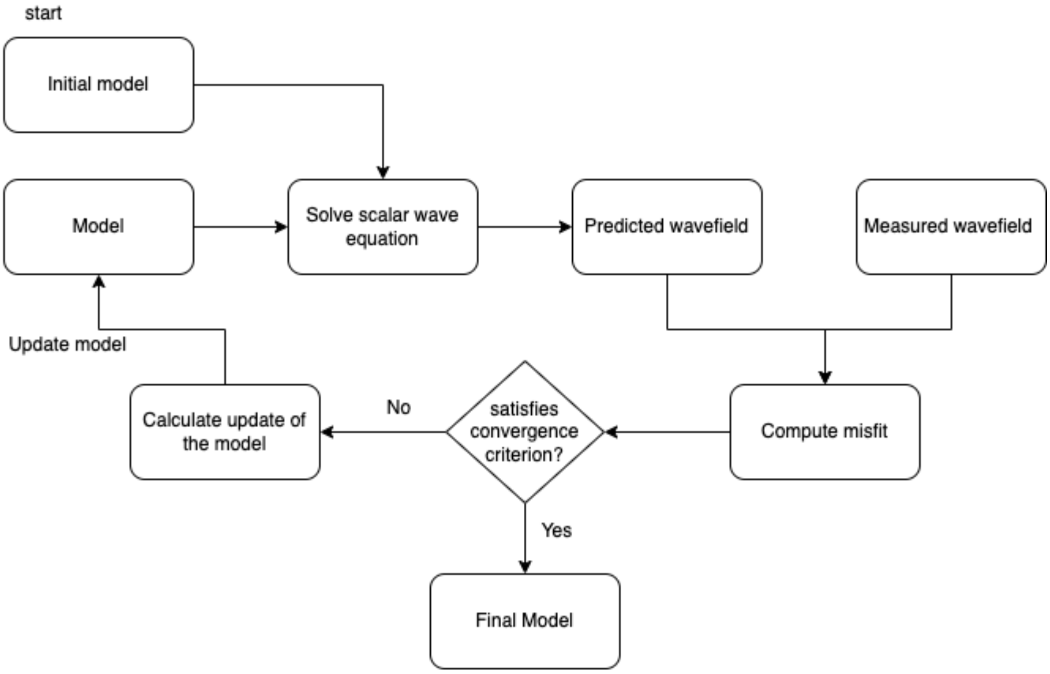


Figure 2.1.: Schematic overview showing the Full Waveform Inversion workflow.

Looking at the figure above, we see that we compute the misfit between the measured values and the predicted wavefield values at every iteration. This misfit is given by

$$J(m) = \frac{1}{2}\int_{\Omega}\int_{\mathcal{T}}\sum_{i=1}^{N_{\mathcal{M}}}\left(u(m;x,t) - u_{\mathcal{M}}(x,t)\right)^2 \cdot \delta\left(x - x_i^r\right) dtdx. \tag{2.1}$$

Equation 2.1 shows the equation we want to minimize in FWI, where $u_{\mathcal{M}}(x_i^r, t)$ is the observed signal for the sensor at $x_i$ and time $t$, and $u\,(m; x, t)$ is the solution of the wave equation used in the experiment, for example, the acoustic wave equation used in Ultra Sonic Non-Destructive Testing (US NDT) [62], or the elastic wave equation used in seismology [61], simulated in the numerical model of the specimen. Moreover, $\Omega$ and $\mathcal{T}$ are the spatial and time domains respectively. At every iteration, we seek to minimize equation 2.1. With this brief introduction to the

Full Waveform Inversion problem, we can identify the two critical ingredients for solving the inversion task: An efficient forward solver to model the wavefield and an efficient gradient computation algorithm to help minimize the high dimensional objective function using a gradient-based optimisation method.

### 2.1.1. Forward problem

The forward problem in FWI comes down to numerically solving the wave equation. Numerical solutions arise from discretisation techniques, where the most popular approach for simulating propagating waves is to use the finite-difference method due to its ease of implementation and understanding [72, 14]. In cases where we need to consider accurate and complex boundary condition cases, for example, when we need to make use of unstructured meshes to implement boundary conditions, we can apply finite element and finite volume methods [12, 34]. Finite element methods offer greater flexibility and accuracy in handling complex boundaries and varying material properties, but they can be computationally more expensive.

With the irruption of AI methods in scientific computing, novel computational methods have arrived. Deep learning and neural networks can approximate solutions to partial differential equations and serve as forward models [16, 45]. However, it is essential to treat using neural networks as forward solvers with caution because neural networks may have problems generalise for data outside their training distribution [81]. There is also the possibility to combine finite difference classical solvers with convolutional neural networks (CNNs) to accelerate simulations by using larger time steps [64]. In section 2.2, we introduce Physics-Informed Neural Networks (PINNs) and explain how these networks yield a solution to the forward problem.

Exploring the literature in FWI reveals that finite difference schemes are the preferred tool for solving the forward problem in FWI when the geometry of the problem is simple enough [62, 61, 20]. Regarding the state-of-the-art papers in FWI, the finite element method is used in [6]. The work presented in this master's thesis employs a second-order explicit finite difference scheme for solving the scalar wave equation implemented in [20].

### 2.1.2. Inverse problem and gradient computation in FWI

As introduced in section 2.1, FWI can be formulated as a non-linear, least squares optimization problem, iteratively solvable through gradient-based optimisation techniques. The goal is to update the model parameters in each iteration to yield a simulated velocity field as close as possible to the measurement data. In other words, we aim to minimize equation 2.1 by updating the parameters $m(x)$. The crucial step in this optimisation procedure is calculating the gradient of the misfit function. Inverse problems in computational science, physics, engineering, and other disciplines can have $10^{10}$ inversion parameters [22]. What is the best approach to compute the gradient of a functional with respect to the model parameters?

An efficient way to calculate the gradient of a function with respect to a large number of parameters is the adjoint state method [17]. Chavent developed this computational method in 1974 to solve inverse problems based on the adjoint state of an equation [7]. It showed massive potential in gradient-based optimisation as it provided a new method that yields the gradient independently on the number of parameters to invert and without calculating the Fréchet derivatives, which can be expensive to compute. Due to its computational advantages, the adjoint state method is considered as the state-of-the-art reference method for classic full waveform inversion [20]. A quick search in the literature of FWI reveals its popularity [6, 62, 73].

The derivation for the gradient calculation using the adjoint state method follows from [6] and [61]. The first step of the adjoint state method requires the formulation of the adjoint state of the optimisation problem. We follow the derivation provided in [6]. Let $\mathbf{L}$ be the partial differential equation operator for the scalar wave equation in heterogeneous medium, and let $\mathbf{u}$ be the solution to this partial differential equation. Considering the misfit function given by equation 2.1, the derivative of the misfit function with respect to a change of the specimen's model in direction $\delta \boldsymbol{m}$ is

$$\nabla_m J\left(m\right)\delta\mathbf{m} = \int_T \int_\Omega \mathbf{u}^\dagger(\mathbf{x},t)\cdot\nabla_m \mathbf{L}(\mathbf{u};\mathbf{m},\mathbf{x},t)\delta\mathbf{m}d\Omega dt. \tag{2.2}$$

The field $\mathbf{u}^\dagger$ is the solution to the adjoint equation. This work focuses on the scalar wave equation and hence operator $\mathbf{L}$ is given by

$$L(u;\gamma) = \gamma\rho_0\ddot{u} - \nabla\cdot\left(\gamma\rho_0 c_0^2\nabla u\right). \tag{2.3}$$

In addition, the scalar wave equation is self adjoint for non dissipative material and thereby, the adjoint equation takes the same form [6]. Setting homogenous boundary conditions $u(\mathbf{x},t) = 0$ and $\dot{u}(\mathbf{x},t) = 0$, the adjoint equation is given by

$$\gamma\rho_0\ddot{u}^\dagger - \nabla\cdot\left(\gamma\rho_0 c_0^2\nabla u^\dagger\right) = f^\dagger, \quad \mathbf{x}\in\Omega, \tag{2.4}$$

with the adjoint volume force given by the integrand of the misfit function.

$$f_s^\dagger(x,t) := -\sum_{i=1}^{N}\left[u(m;x,t) - u_{\mathcal{M}}(x,t)\right]\delta\left(x - x_i^r\right). \tag{2.5}$$

The adjoint problem is given by:

$$\begin{aligned} \gamma(\mathbf{x})\rho_0 u^\dagger_{tt}(\mathbf{x},t) - \nabla \cdot \left(\gamma(\mathbf{x})\rho_0 c_0^2 \nabla u^\dagger\right) &= f^\dagger, \quad \mathbf{x} \in \Omega,\ t \in [0,T], \\ u^\dagger(\mathbf{x},T) = u^\dagger_t(\mathbf{x},T) &= 0, \quad \mathbf{x} \in \Omega, \\ u^\dagger &= 0, \quad \mathbf{x} \in \partial\Omega. \end{aligned} \tag{2.6}$$

Using the definition of the derivative with respect to $\gamma$, substituting the given expression into equation 2.2 and integrating by parts, the derivative of the misfit function with respect to the parameters is given by:

$$\nabla_\gamma J \delta\gamma = \int_T \int_\Omega \left[-\rho_0 \dot{u}^\dagger \dot{u} + \rho_0 c_0^2 \nabla u^\dagger \cdot \nabla u\right] \, dt \, d\Omega, d\gamma. \tag{2.7}$$

In the work presented in this master's thesis, we make use of the adjoint method for computing the gradient of the misfit function when attempting to solve the FWI problem. The implementation is performed using PyTorch and we use the same forward solver to yield the solution to the scalar wave equation, as well as the solution to its adjoint equation.

As mentioned, the adjoint state method is the state-of-the-art method for computing gradients in FWI. However, researchers have attempted to solve the FWI problem by computing the gradients differently. For example, it is possible to use automatic differentiation implemented in deep learning frameworks like PyTorch and TensorFlow to calculate the gradient of the loss function with respect to the material parameters when using a neural network ansatz to compute the material distribution of the specimen. This exactly is done and showcased in [20]. The results obtained for the FWI problem are accurate but computationally inefficient. The article states that the computational time per epoch dropped from 2.46s to 1.05s when exploiting the adjoint state method compared to automatic differentiation. Nevertheless, the article proves automatic differentiation also works well.

### 2.1.3. Full Waveform Inversion in seismology

To understand the need for the development of FWI, it is helpful to briefly explain the main milestones in the history of seismology. We can consider the history of seismology to start in the mid-18th century when J. Michell associated earthquakes with wave propagation at the surface of the Earth [44]. The theory of elasticity developed throughout the 18th and 19th could demonstrate the observations stated by J. Michell. Two potential solutions, namely P and S waves propagating with unequal velocities, were suggested analytically for homogeneous and unbounded media. Regarding arbitrarily heterogeneous media, solutions have only become available in recent years as a result of advancements in computational science. Furthermore, what could be the first experiment using active sources for investigating wave propagation was performed by father and son Mallet using gunpowder explosions to produce perturbations. Not only did they possibly perform the first active source experiment, but they eventually also resolved a seismic inverse problem as they related wave speed variations and material property variations.

After a devastating earthquake in 1897, Richard Oldham identified the two analytical solutions, P and S waves, traveling at different speeds proposed for the elastic wave equation [47]. After identifying P and S waves, Oldham discovered the fast decay of the amplitude in P waves and inferred the existence of the Earth's outer core [48]. From Oldham's work, B. Gutenberg determined the radius of such outer core [19]. Other significant milestones in the history of seismology include the discovery of two different types of P-waves and S-waves by A. Mohorovičić, Lehmann's inference of the Earth's inner core in 1936, and advances to infer complete radially symmetric Earth models by compiling traveltime tables [27]. During the 20th century, seismologists and physicist refined the existing Earth models, and discoveries of 3D heterogeneities were possible using tomography methods. Some examples of the research conducted to model the Earth and its subsurface heterogeneities include [1, 29]

.

All these scientific milestones were possible thanks to the simple representation of seismic waves by rays. The spread of ray theory-based approaches in seismology was motivated by its simplicity, its simple analysis of results, and the fact that obtaining solutions to the PDE describing wave propagation (scalar wave equation) in heterogeneous media proves difficult. However, the simplifying assumptions used compromise the accuracy of the resulting Earth models. Ray theory's capabilities in seismology have been a source of distress at the end of the 20th century [76, 66]. Ray theory is applicable when the length scales of 3D heterogeneities are significantly smaller than the dominant wavelength, imposing an upper bound for the quality in terms of the resolution of the tomographic images arising from the simplifications derived from ray theory [13]. Since the 2000s, the scientific community has been striving to develop new methods surpassing imposed limitations. Such developments encompass finite-frequency tomography and full-waveform inversion [79, 15, 2, 58, 68].

Main discoveries in modern seismology were achieved by studying travel time information. Travel time in seismology refers to the time seismic waves travel from the focus point to a certain point in the Earth's crust. The development of Global Seismic Networks, allowing the collection of large amounts of quasi-real-time digital open-source data, catapulted advancements in seismology to develop more accurate Earth models through data-driven techniques and the study of normal modes. Scientists see normal modes in seismology as seismic wave patterns revealing insights into the Earth's structure.

There were different methods at the time to develop the best subsurface models possible. Seismologists used differential seismograms to conduct high-resolution upper mantle tomography [77]. Differential seismograms are a seismological tool used to understand the response of wave propagation to small changes in an Earth model. The standard procedure used the Born approximation to obtain differential seismograms, simplifying wave propagation modeling and representing the differences between the recorded data and the data obtained after perturbing the initial model. The procedure involves comparing several differential seismograms and analysing how changes in the model parameters affect the resulting synthetic data, thereby helping refine the initial Earth model. The differential seismograms are used to yield a perturbed model, which is used to generate synthetic seismic data. This synthetic data is compared with recorded data to estimate the sensitivity matrix. The sensitivity matrix describes the degree of sensitivity of the synthetic data with respect to the model parameters [28]. Partial derivatives of the generated data with respect to the model parameters are calculated to estimate the sensitivity matrix. This is required before the inversion takes place, which relies on the normal mode description to allow the optimization algorithm realisable, by explicitly approximating the Frechet kernel even though considering a large number of seismograms.

At the same time, efforts were made to provide high-resolution imaging using data recorded at different depths below the Earth's surface and from various angles. Using the abovementioned technique and explicitly estimating the sensitivity matrix becomes too expensive. To overcome this difficulty, scientists turned to the exploding-reflector concept, yielding high-quality resolution imaging of the subsurface after some physical considerations were implemented to better account for how waves propagate through the ground [8]. The idea behind this method lies in carefully adjusting the recorded seismic data and aggregating the corrected seismograms to obtain a detailed subsurface image. This approach, known as migration, overcomes the difficulties from building the sensitivity matrix but introduces other challenges.

In particular, data collected by seismic reflection surveys do not collect data from various angles, and the sources used have a limited frequency bandwidth. As a result, Earth structures corresponding to medium-sized wavelengths need to be better represented in the data [26]. The workaround is a twofold procedure: Firstly construct a physical model of the Earth structure considering wave propagation in the ground. Moreover, polish and refine the resulting model to increase accuracy by different migration means or iteratively updating the physical model [4, 65]. These solutions, however, offer a poor sensitivity to the large and medium-sized subsurface wavelengths. In addition, Tarantolla and Lailly proposed an iterative procedure improving Claerbourt's migration imaging technique. This procedure reformulated the migration imaging approach into a local optimization problem, minimizing the squared error between the recorded and synthetic measurements in a gradient-based fashion. Each iteration adds the perturbation found through gradient descent to the initial model, improving the accuracy in each iteration.

We can use various forward modeling approaches to achieve waveform-fitting. Some examples include: reflectivity techniques in layered media, finite differences, finite element methods, and adaptation of ray theory. These methods modeled the forward problem correctly but encompassed high computational resources. A less computationally demanding approach connects ray theory general Radon reconstruction techniques and least squares optimization [39]. Because Green's functions are computed in smoothed media with the ray theory, we can use the Born approximation to linearise the forward problem, and the optimization is iterated linearly, which means the background model remains the same over the iterations [73]. These imaging techniques are commonly known as migration or inversion methods.

Contrary to this, FWI considers the total information content of the seismogram for the optimization procedure. At every iteration, we model the full wave equation in the final model state of the previous iteration [73]. FWI considers all types of waves, making it a general, comprehensive and robust approach to seismic imaging.

### 2.1.4. Full Waveform Inversion in Non-Destructive Testing

During the 20th century, the increase in engineering projects and new infrastructure has brought along a growing necessity to test, monitor and analyse the state of infrastructure and its constituting components, partly for safety reasons, and thereby an incentive to focus on developing non-destructive testing techniques [80, 43, 67].

Non-Destructive Testing is used to build a model of the internal structure of a component in terms of its material composition and geometric features without inducing any physical or chemical change to it. High demand for accurate, detailed, and efficient NDT imaging has emerged to ensure high-quality standards remain high and consistent in industrial production processes. In the last decades, several NDT methods have emerged that strive to provide high-resolution images of the physical systems encountered. These techniques include, and are not limited to, infrared NDT, radiography testing, and ultrasonic NDT [51, 75, 62].

Classical US NDT uses elastic waves to infer the material distribution of a specimen without causing any impurity to it. The idea is to propagate a low-amplitude and high-frequency sound disturbance through the specimen, taking measurements of travel time and wave intensity at specific collocation points in the specimen. The frequency range of waves in a US NDT setting is 500 KHz to 20 MHz. Flaws in the material will affect the measurements taken at the sensors due to reflective effects. The collected measurements are then compared with benchmark results of flawless specimens. This approach allows the detection of flaws in a material resulting from a discrepancy between the recorded measurements in the tested specimen and the benchmark measurements of the flawless material. However, no additional information regarding location, orientation, and dimension can be retrieved. Figure 2.2 depicts the classical procedure in NDT.

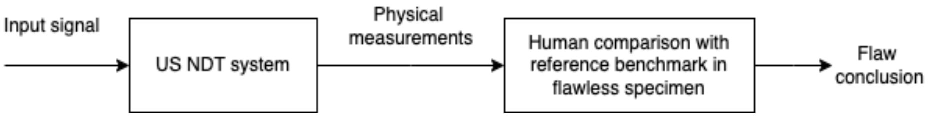


Figure 2.2.: Standard system used in NDT. The diagram can be found in [62].

Improvements of classical NDT lead to model-based NDT, which allows not only to determine whether flaws exist in the specimen but also to characterise the geometry and dimensions of the flaws by numerically comparing signal characteristics of the measured data and simulated data. Widely used model-based NDT methods include SIRT and ART [40, 46]. Figure 2.3 depicts a model-based NDT procedure. FWI can be employed to accurately determine the parameters of complex earth models in seismology. Consequently, with some adaptation to the more minor scales used in non-destructive testing, FWI holds significant potential for such experiments. The idea of FWI in NDT is to use an adaptable model of the flawless specimen that will be iteratively updated to match the reference measurements instead of having a fixed reference specimen. Starting from an approximative model obtained from a CT scan, we adapt the initial model in an iterative fashion, updating the parameters to hopefully converge to a model that yields the same signals at the sensors as the measurements initially taken. This technique is an improvement compared to previous methods used in NDT, allows the production of accurate models of the material, and better characterises the detected flaws. Applications of FWI in US NDT include simultaneous detection of multiple cracks, flaw characterisation, and other applications. Some efforts have been made already to explore the potential of FWI in the field of NDT [56, 59].

In FWI, we consider two main steps. First, the mathematical modeling of the specimen, and secondly, and the numerical simulation of the wavefield in the specimen. This task is denoted as the forward problem. Once we numerically solve the wave equation to yield an accurate velocity model, we now want to find the material parameters that yield numerical results similar to the measurements made. This poses the inverse problem we will consider in this work.

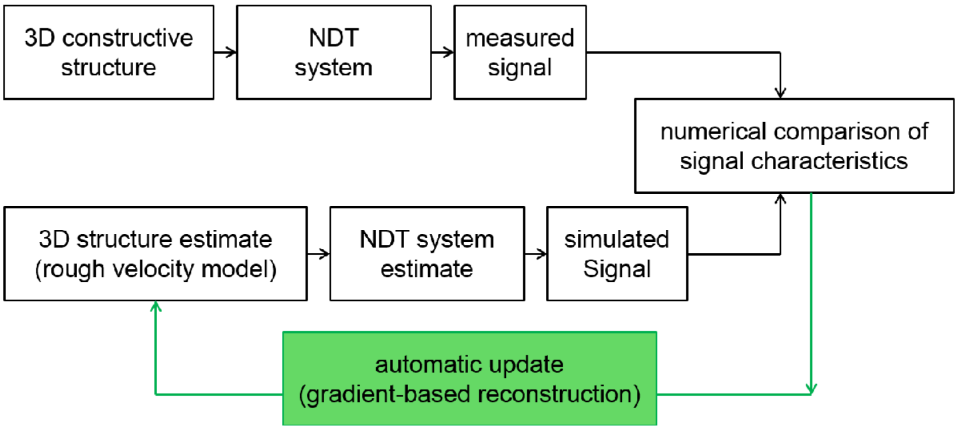


Figure 2.3.: Algorithm used in model based NDT. The diagram can be found in [61].

## 2.2. Physics informed neural networks

Physics-Informed Neural Networks (PINNs) are machine learning algorithms capable of solving problems described by partial differential equations. What differentiates PINNs from other NNs is the peculiarity that PINNs embed the governing physics equations of the system into the training process through the loss function of the neural network [52]. The loss function comprises different terms, which account for initial and boundary conditions and the PDE residual. These modifications made to the loss function make PINNs more powerful when solving physics problems than classical machine learning methods, mainly when operating in low data regimes [20]. In this small data regime, PINNs stand out compared to classical machine learning algorithms, which offer low guarantees for convergence when trained with limited data. Classical machine learning methods are purely data-driven and do not include domain expertise in the learning process. This is why achieving a good generalisation is difficult, especially when exposed to small amounts of data. PINNs provide an alternative, improved method to classical machine learning techniques when analyzing complex physical systems, as we can embed prior knowledge in the training process. The PINN algorithm offers a meshless alternative allowing to approximate the solution of the partial differential equation describing the problem by transforming the initial value problem into an optimisation problem. Embedding the governing equations and physical constraints into the loss function acts as a penalizing term, restricting the space of plausible solutions.

PINNs appeared in 2017 for the first time to revolutionise research in scientific computing through their smart embedding of prior domain knowledge in the training process. However, the idea of including prior domain knowledge into a machine learning algorithm goes back some years before the birth of PINNs. In his research, Owhadi laveraged prior domain knowledge about the solution spaces of PDEs using a bayesian inference formulation of the problem, to identify bases for the numerical homogenization of arbitrary integro-differential equations [49]. Furthermore, another example where such domain expertise was exploited was to use a Gaussian process regression to approximate linear and non linear operator functionals, as well as infer solutions and yield uncertainty estimations for the 1d time-dependent advection–diffusion–reaction, the Poisson equation in ten dimensions, and fractional sub-diffusion[55, 54, 53]. Furthermore, during the 90s, several research papers appeared aiming at leaveraging prior knowledge to solve partial differential equations [36, 50, 11]. At first, these advancements only considered problems with regular borders, but this was later extend to account for arbitrary geometries [37].

Advancements in technology leading to an increase in computing power, new specialised hardware development, new methodologies for training networks, and the introduction of deep learning frameworks such as TensorFlow and Pytorch providing automatic differentiation capabilities, boosted research in using NN for solving PDEs. Further research into the state of the art deep learning architectures were part of the last steps towards the development of the PINN algorithm [41, 35]. The release of the paper by Raissi et al introducing Physics Informed Neural Networks marks a milestone in physics informed machine learning and opened the door for research in physics informed surrogate models [52, 32]. As a small remark, the paper showcases the potential of PINNs for dealing with two types of problems. Firstly, data driven solution of PDEs, and secondly, data driven discovery of partial differential equations. This thesis will deal with data driven discovery of partial differential equations [60]. In other words, we will compute an unknown function involved in the considered PDE from measurements.

### 2.2.1. Problem Set Up and Model Estimation

Many physics laws are described by parametrised non-linear partial differential equations in the form of

$$\frac{\partial u}{\partial t} + \mathcal{N}[u; \lambda] = 0, x \in \Omega, t \in \mathcal{T}. \tag{2.8}$$

In equation 2.8, $u(x,t)$ denotes the solution in the interval $t \in [0, T]$ and at a point $x \in \Omega$ in space. In addition, $\mathcal{N}[u; \lambda]$ denotes a non-linear differential operator with coefficients given by $\lambda$. PINNs can be exploited to solve two types of tasks. Firstly, PINNs can be used for data-driven inference. Data-driven inference refers to solving the forward problem, i.e finding a solution to the partial differential equation. Secondly, PINNs can be further used for data-driven identification of partial differential equations. In other words, this accounts to solving an inverse problem.

Considering the most general form, PINNs have the potential to solve both the forward and inverse problems defined as

$$\begin{aligned} \mathcal{F}(u(z);\gamma) &= f(z) \quad \text{for } z \text{ in } \Omega, \\ \mathcal{B}(u(z)) &= g(z) \quad \text{for } z \text{ in } \partial\Omega, \end{aligned} \tag{2.9}$$

where on the one hand, the forward problem accounts for finding the function $u$ for every $z$ in $\Omega$ where $\gamma$ are physical parameters and $z$ are space-time coordinates, and on the other hand, the inverse problem accounts for approximating $\gamma$ from the data. The right hand side of equation 2.9 gives the function representing the data, both in the interior of the domain as in its boundary. Moreover, $\mathcal{F}$ is the non linear differential operator. As for every inital value problem, we need initial and boundary conditions to solve it. $\mathcal{B}$ represents these initial and boundary conditions. Regarding the forward problem, a neural network is used to approximate the solution of the partial differential equation, where the model is defined by trainable parameters $\theta$, where

$$\hat{u}_\theta\left(z\right) \approx u\left(z\right). \tag{2.10}$$

The vector $\theta$ represents the vector of the model parameters for both the forward and inverse problems, given that PINNs can resolve both tasks [33]. As mentioned earlier in this work, the PINN algorithm can be formulated as an optimization problem in which the governing mathematical equations of the system are embedded in the loss function of the network that we seek to minimize during training. Mathematically speaking,

$$\theta^* = \arg\min_\theta \left(\omega_\mathcal{F}\mathcal{L}_\mathcal{F}(\theta) + \omega_\mathcal{B}\mathcal{L}_\mathcal{B}(\theta) + \omega_d\mathcal{L}_\text{data}\left(\theta\right)\right) \tag{2.11}$$

where $\mathcal{L}_\mathcal{F}(\theta)$, $\mathcal{L}_\mathcal{B}(\theta)$, and $\mathcal{L}_\text{data}\left(\theta\right)$ are the partial equation, initial and boundary conditions, and measurements losses. Each of the losses is weighted with its corresponding weight to counteract for the differences in magnitude of the different loss terms. In this section of the thesis I just describe the general problem set up, and will later define the exact losses, intrinsic to the problem we will study. As a last remark, I want to note that when considering the forward problem, PINNs can be regarded as an unsupervised learning algorithm, however, when looking at the inverse problem, PINNs can be considered a supervised learning algorithm [9].

### 2.2.2. NN architectures used in PINNs

The universal approximation theorem states that any continuous function can be arbitrarily approximated up to a sure accuracy by a multilayer perceptron with one hidden layer and a finite number of neurons [24]. However, different network architectures perform differently, and choosing the suitable model can be challenging. We can classify NN into two broad groups: shallow and deep networks. The first group consists of networks with one or a few hidden layers, whereas the latter have many. Regarding the PINN framework, the majority of implementations rely on feed-forward neural networks, although there are also cases employing other architectures such as CNN, RNN, and GAN [20, 71, 78].

Assuming we opt for the general implementation of PINNs using feed-forward neural networks, how many hidden layers and neurons per layer yield the best solution? Experiments and research suggest that each problem requires its own, particular architecture, and there is no magic, general architecture working for all problems [70, 30, 74].

## 2.3. PINNs for Full Waveform Inversion

Ever since the advent of PINNs, there has been great recognition from the scientific computing world. As we have seen in the sections above, PINNs can be used to approximate solutions to PDEs and jointly solve forward and inverse problems. When used to find forward solutions to PDEs, they are computationally more expensive than classical partial differential equation solvers [42]. However, PINNs show great potential for solving Inverse problems, which is the main focus of this work. As the thesis title indicates, we focus on the problem of full waveform inversion, where the aim is to find the material distribution of a specimen from wavefield measurements taken at defined collocation points.

Different research papers tackle the problem of full waveform Inversion in different domains, in particular seismology and non-destructive testing. Following the work in [20], the problem under study dictates the choice and implementation of the PINN and the loss function chosen accordingly. We can encounter two types of inverse problems when performing FWI, in terms of the domain's knowledge, which influences the choice of PINN. We make a distinction between inverse problems with full domain knowledge of the solution field and inverse problems with partial domain knowledge of the solution field, following the approach found in [20].

In the first case, the velocity field $u\left(\vec{x}, t\right)$ is known everywhere in the domain, hence approximating the velocity field with a neural network is not needed. This paper provides the state-of-the-art for solving this inverse problem using PINNs [63]. The paper presents an optimized PINN, trained to identify and characterise flaws inside a metal plate, seeking to solve problems in wavefield imaging at ultrasonic frequencies. The study focuses on the acoustic wave equation:

$$u_{tt} = v^2(x,y)\Delta u, \quad x \in \Omega, \text{ and } t \in [0,T], \tag{2.12}$$

a second order linear PDE, governing the propagation of acoustic waves in a medium, where $u\left(t,x,y\right)$ is the solution, $v\left(x,y\right)$ is the sound speed and $\Omega \in \mathbb{R}^2$. In their research, scientists train a neural network to approximate the solution to the wave equation, and the residual of equation 2.12 is embedded in the loss function. The physics-informed neural network implemented for this study is trained with a loss function containing two terms: the misfit between the measurement data and the approximation and the residual of the PDE. Moreover, they use a fully connected network with 96 neurons per layer and a depth of 4 layers. Furthermore, the gradient of the loss function with respect to the model parameters is computed using automatic differentiation [3]. In addition, the optimization problem can be solved using any form of gradient descent algorithm. In particular, the ADAM optimizer was used in this work [31].

In this work, researchers encountered slow convergence or no convergence of the loss function during training when using $\tanh$ as an activation function [63]. The choice of loss function is intrinsically related to the problem being studied, and there is no criterion to help us with this choice. To circumvent the convergence problems, an adaptive activation function scheme was implemented [25]. The authors included an extra hyper-parameter to multiply the loss function and input this product into the activation function in this method. The parameter is also optimized in the training loop through a gradient-based procedure. Finally, a last modification needs to be done to account for the effect of the learning rate. The parameter introduced is multiplied by a constant much larger than 1. The introduced parameter was seen to steepen the activation function, helping with faster convergence. This article by Japtap et al. showcases the effect of $a$ in convergence in a selection of machine learning problems [25].

Regarding the second type of inverse problems, the state-of-the-art is showcased in [57]. In this research project, the authors propose a new approach to model wave propagation and full waveform inversion using PINNs. This new approach utilizes two PINNs, one for the forward modeling and one for the inverse problem. Both networks are fully connected, with the second network being much smaller than the network used to compute the solution to the forward problem due to the variations in the specimen's material distribution being less complex than the varying wavefield. Different computational experiments were performed, evaluating the PINN algorithm for forward modeling of the acoustic wave equation and inverse modeling to find the speed wave speed distribution in the material. Regarding inverse modeling, a study is conducted to evaluate the PINN's capabilities to recover sharp 2-D anomalies from a synthetic Crosswell experiment. The implemented PINN successfully identifies and characterises the anomaly in terms of location, orientation, and magnitude. It is important to note that the solution retrieved by the PINN is smoothed out compared to the sharp discontinuity of the anomaly in the grand truth distribution. The authors suggest that reducing the smoothening effect of the inversion is plausible if a larger training data set was used for the pde residual in the loss function and using the L1 norm for the minimization.

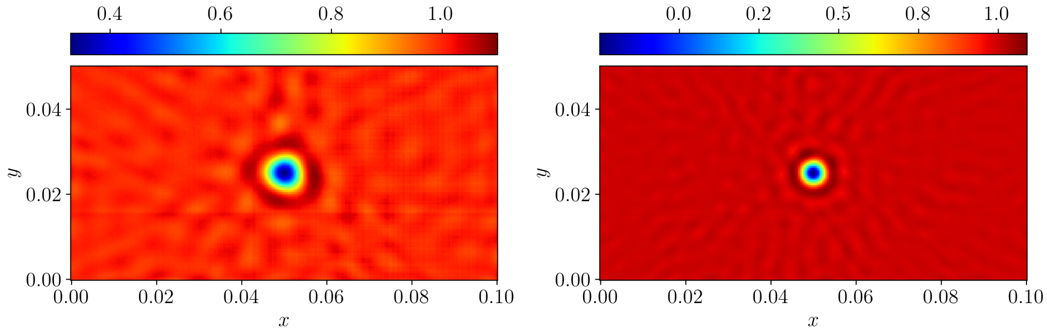
## 2.4. Weight initialization and SWIM network

An essential step in the recipe for training neural networks is initializing weights and bias terms. Weight initialization has a noticeable impact in the convergence rate, accuracy, and generalization of the model [10]. The choice of initialization affects the loss landscape of our model. It is known that increased diversity in the initial weights favours a good starting point in the loss landscape, making it easier for the moment to escape local minima and hopefully converge to the global minimum. Contrary to this, homogeneity leads to the model getting stuck in local minima and failing to converge to the optimal solution. Several weight initialisation techniques exist to tackle this problem, such as Xavier-Glorot initialization, random initialization, and He initialization. Xavier-Glorot initialization considers the number of inputs and outputs of each layer and seeks to keep the variance constant throughout each layer of the neural network. Moreover, He initialization was introduced to consider non-linear activation functions while keeping the variance through a layer constant. For general cases, Xavier-Glorot initialization is the dominating approach. However, the optimal weight initialization choice depends on the problem we want to tackle and the model's architecture [10].

Training neural networks is often a synonym of minimising the loss function with respect to the network's parameters, and optimising the weights and biases through gradient descent techniques. With the aim of reducing the inference time to solve the inverse problem faster, we explore new methods of sampling the weights of a feed forward neural network. We closely follow the work in [5] to sample the weights of our hidden layer, in efforts to reduce the number of parameters being trained and converge faster to the solution. We make use of sampled networks, where weights and biases in hidden layer neurons are entirely determined by two points in the input space. After constructing all hidden layers in this way, the coefficients and bias term of the linear output layer are determined by through optimisation techniques.

Formally, let $\Phi$ be a feed forward neural network constructed with $L$ hidden layers, composed of parameters $\left\{W_l, b_l\right\}_{l=1}^{L} + 1$ and activation function $\phi_l$. Rigorously, we can write the output of each layer as $\Phi^{(l)} = \phi^l\left(W_l\Phi^{l-1}\left(x\right) + b_l\right)$. In addition, let $N^l$ be the number of neurons in the $l$th layer. The formal mathematical definition of sampled networks can be found in paper [5].

The idea of using the weights and biases from a network that does not need an iterative training based on gradient descent looks very promising at first sight to speed up the inference time when solving the inverse problem. However, is the SWIM network capable of fitting the heterogeneous material distribution of a specimen in the context of NDT? Before performing any experiments and working on the inverse problem we perform a sanity check to see if the weights and biases obtained from the SWIM algorithm can be used to obtain the grand truth material distribution. The SWIM network can be configured for two sampling algorithms: uniform and non uniform sampling. The respective approximations can be found in figures 2.4 and 2.5.

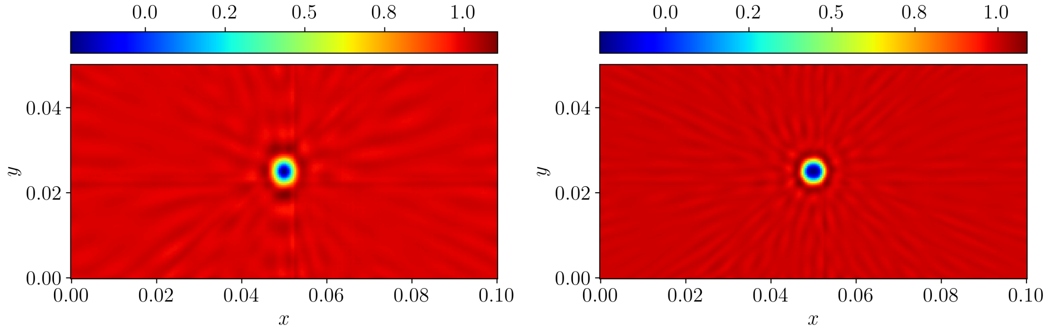


(a) Hidden layer with 1000 neurons.

(b) Hidden layer with 10000 neurons

Figure 2.4.: Results obtained after fitting the ground truth material distribution with the SWIM network for 1000 and 10000 neurons hidden layer using uniform sampling.

The SWIM network used to fit the ground truth material distribution in paper [20] contains one hidden layer with 10000 neurons. The weights of this network can be sampled using two different sampling algorithms. The simplest algorithm is uniform sampling, in which pairs of values inside the domain are sampled uniformly. Furthemore, weights can also be sampled using the SWIM algorithm described in [5]. The algorithm samples data points in areas where there are high gradients, capturing in this way the more important variations in the function.



(a) Hidden layer with 1000 neurons

(b) Hidden layer with 10000 neurons

Figure 2.5.: Results obtained after fitting the ground truth material distribution with the SWIM network for 1000 and 10000 neurons hidden layer with non-uniform sampling.

In addition, the network can be configured with additional parameters that influence how the network behaves. For example, there exists a parameter that increases our chances of sampling different data pairs, and hence capture more areas with high gradients, yielding a better approximation of the function. This parameter is called `repetition_scaler`. The code provided in source code 2.1 shows how a network can be set up with the SWIM network implementation, and showcases important features. To fit a function we need the input data and output data, given by `data_in` and `data_out` respectively. In the case of uniform sampling, the output data can be arbitrary, but of the same shape as the input data, because the actual values of the output data are not taken into account. We just sample pairs uniformly. Contrary to this, when sampling with the SWIM algorithm, the output data is important because the algorithm will select pairs of the form (input data, output data) that yield high gradients. Moreover, we can select the activation function used after each hidden layer, as well as whether we are tackling a classification or regression problem. Furthermore, hidden layers are `Dense` objects, whereas the output layer is a `Linear` object. The `regularization_scaler` parameter plays a role for optimising the linear output coefficients in the case where there is not a unique answer and we need to get the optimal solution by minimising the least squares error. Moreover, the network is assembled with a list, named `steps` in the code, containing all the hidden layers and output layer of the network. In source code 2.1 we cannot see the output layer when building the output layer because we build this ourselves in experiments that follow. The objective of including the source code is to give a short introduction and overview of the parameters that can be configured in the network.

```python
# SWIM NETWORK
data_in = xy
data_out = gammam.numpy().ravel().reshape(data_in.shape[0],)
print("data out shape = ", data_out.shape)
k =  10_000
basis_hidden_1 = Dense(
    layer_width=k,
    activation=np.tanh,
    parameter_sampler="tanh",
    sample_uniformly=True,
    random_seed=38,
    is_classifier=True,
    repetition_scaler=100
)
basis_output = Linear(
    regularization_scale=1e-8
)
steps = [
    ("hidden-1", basis_hidden_1),
]
basis = Pipeline(steps=steps, verbose=False)
basis.fit(data_in, data_out)
```

Source Code 2.1.: Python code for setting up and training the SWIM network.

With this gentle and brief introduction into the SWIM algorithm we have now addressed all the necessary points for understanding the work in the experiments that will follow. I hope you enjoy the read.

# 3. Full Waveform Inversion with Neural Network based Ansatzfunctions

## 3.1. Problem description

Full waveform inversion (FWI) seeks to model a specimen's material distribution in a non-destructive and non-invasive manner. In this work, we initially consider a 2D rectangular domain with a circular aperture inside the domain. At first, the void will have a circular shape, as shown in the diagram below, but we will also study the problem with voids of different shapes. This study aims at solving the inversion problem and identify and characterise the void in the domain with a shallow feed-forward neural network. Later, following experimental results of approximating the discontinuous material distribution function, we evaluate the capabilities of different architectures and easier material distributions when solving the FWI task.

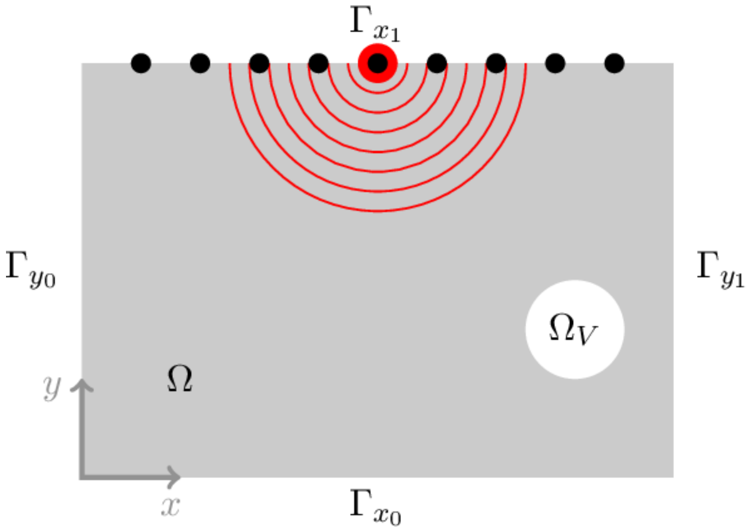


Figure 3.1.: Representation of the problem in US NDT showing a rectangular domain $\Omega$ with a circular void $\Omega_V$. The boundary of the domain $\partial\Omega$ is given by $\Gamma_{y_0} \cup \Gamma_{y_1} \cup \Gamma_{x_0} \cup \Gamma_{x_1}$. The diagram is taken from [20].

In FWI, the material distribution is approximated by using sources that emit a wave signal in the specimen, and sensors that record the displacement at specific points in the domain (these points are called collocation points). The governing equation for this problem is the scalar wave equation in an isotropic heterogeneous medium, given as

$$\rho(\boldsymbol{x})u_{tt}(\boldsymbol{x},t) - \nabla \cdot \left(\rho(\boldsymbol{x})c(\boldsymbol{x})^2 \nabla u(\boldsymbol{x},t)\right) - f(\boldsymbol{x},t) = 0 \quad \text{on } \Omega \times \mathcal{T}. \tag{3.1}$$

where $u(\boldsymbol{x},t)$ is the solution to equation 3.1, $\rho(\boldsymbol{x})$ is the density of the material, $c(\boldsymbol{x})$ is the wavespeed, and $f(\boldsymbol{x},t)$ is the volume force. Moreover, the initial and boundary conditions used are homogeneous Neumann, represented as

$$\begin{aligned} &u_x(\boldsymbol{x},t) = 0 \quad \text{on } \Gamma_y \times \mathcal{T}, \\ &u_y(\boldsymbol{x},t) = 0 \quad \text{on } \Gamma_x \times \mathcal{T}, \\ &u(\boldsymbol{x},0) = u_t(\boldsymbol{x},0) = 0 \quad \text{on } \Omega. \end{aligned} \tag{3.2}$$

In addition, we parameterise the inverse problem as done in [20]. We use an indicator function $\gamma$ which scales the density, such that the intact material has a density of $\rho_0$ and $\gamma(\boldsymbol{x})$ is bounded between the lower bound $\epsilon$, and the upper bound 1, with $\epsilon \ll 1$ and greater than 0:

$$\rho(\boldsymbol{x}) = \gamma(\boldsymbol{x})\,\rho_0. \tag{3.3}$$

Finally, the wave speed is assumed to be constant $c(\boldsymbol{x}) = c_0$, to yield the parameterised scalar wave equation we will use in our study:

$$\gamma(\boldsymbol{x})\rho_0 u_{tt}(\boldsymbol{x},t) - \nabla \cdot \left(\gamma(\boldsymbol{x})\rho_0 c_0^2 \nabla u(\boldsymbol{x},t)\right) - f(\boldsymbol{x},t) = 0 \quad \text{on } \Omega \times \mathcal{T}. \tag{3.4}$$

Furthermore, we need to model the volume force emitting the signal. We consider four sources placed in four locations on the top boundary of the specimen. Each source is considered to be a point source, modeled using a time varying excitation $\psi(t)$ and a spatial Dirac delta function,

$$f(\vec{x},t) = \psi(t)\,\delta(\boldsymbol{x} - \boldsymbol{x_s}), \tag{3.5}$$

with the time varying excitation given by

$$\psi(t) = \begin{cases} \psi_0 \sin(\omega t) \sin\left(\frac{\omega t}{2n_c}\right), & \text{for } 0 \le t \le \frac{2\pi n_c}{\omega} \\ 0, & \text{for } \frac{2\pi n_c}{\omega} < t. \end{cases} \tag{3.6}$$

The goal of solving the inverse problem is to retrieve the specimen's material distribution, given by the predicted function $\hat{\gamma}(\boldsymbol{x},\boldsymbol{m})$ and parameterized by $\boldsymbol{m}$, that yields the solution to the wave equation 3.4, which is closest to $u_{\mathcal{M}}(\boldsymbol{x}_i,t)$, after solving it with the current state of the parameters. The inversion is posed as an optimization problem, where the misfit given by

$$\mathcal{L}(\hat{\boldsymbol{u}},\boldsymbol{u}_{\mathcal{M}}) = \frac{1}{2}\int_{\Omega}\int_{\mathcal{T}} \sum_{i=1}^{N_{\mathcal{M}}} \left(\hat{u}(\boldsymbol{x},t;\boldsymbol{m}) - u_{\mathcal{M}}(\boldsymbol{x}_i,t)\right)^2 \cdot \delta(\boldsymbol{x} - \boldsymbol{x}_i^r)\, d\mathcal{T} d\Omega, \tag{3.7}$$

is the function we seek to minimize by updating $\boldsymbol{m}$. We will follow a gradient based optimization procedure, using ADAM optimizer. To compute the gradient of the misfit function with respect to the model parameters, $\nabla_m \mathcal{L}_{\mathcal{M}} \left( \hat{\boldsymbol{u}}(\boldsymbol{m}), \boldsymbol{u}_{\mathcal{M}} \right)$, we will employ the adjoint state method for the main work of the thesis. Moreover, results from [20] will also be included and used as a benchmark for comparisson with our results.

The main objective of this thesis is to provide a detailed study of how well can a shallow feed-forward neural network solve the inversion problem in the context of ultrasonic non destructive testing, and develop a new hybrid method that exploits the computational advantages of the adjoint state method for gradient computation w.r.t to the inversion parameters, and a novel weight sampling technique to speed up the inference time needed to solve the inverse problem without the need to train the hidden layer weights and bias terms. Ideally, we would like to solve the FWI problem with the architecture described in section 3.2, and the ”Sample Where It Matters” (SWIM) weight sampling algorithm described in section 2.4. Moreover, we will also study the capabilities of other architectures to solve simpler FWI problem, following experimental results of the main objective of this master’s thesis.

## 3.2. Neural network architecture

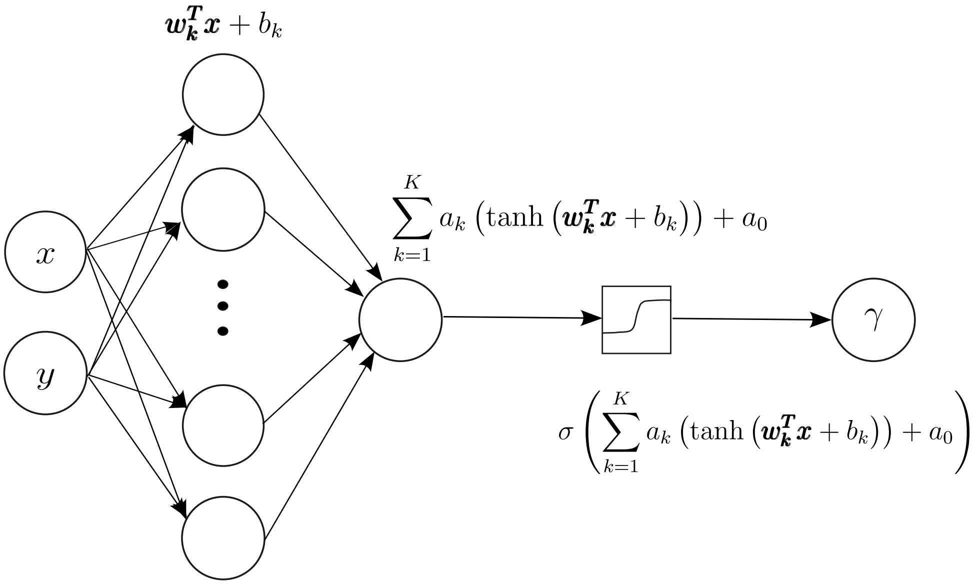


Figure 3.2.: Neural network architecture used for our proposed FWI method. Inputs are 2D coordinates. The network is made of one hidden layer with $\tanh$ activation functions, one output linear layer, and a sigmoid activation function after the output to squeeze the output of the network between 0 and 1.

In this master's thesis, the ultimate goal is to solve a FWI problem, that is, approximate the material distribution of a specimen. FWI requires to solve the wave equation at every iteration of the optimization loop. Thereby, we need to discretize the domain to find a solution to equation 3.4 at every point in our grid. We use the same discretization to solve the inverse problem, and want to find the specimen's density at every gridpoint in our mesh. Consequently, the input of the neural network will be individual 2-d equidistant gridpoints, resulting from the discretization of our domain. Moreover, the SWIM algorithm allows us to sample weights and biases for fully connected neural networks, and thereby, we explore fully connected networks for solving the inverse problem. We use a feed-forward neural network with one hidden layer. We use $tanh$ as an activation function after the hidden layer, and we include a sigmoid after the output layer to ensure the output is between 0 and 1, agreeing with the parametrization of voids in the problem description. Moreover, after the sigmoid activation function, it is essential to scale the output to the range $[\epsilon, 1]$ to ensure the numerical stability of the scalar wave equation's solution. Moreover, we observe the approximations made by the SWIM network when fitting the ground truth function in figures 2.4 , 2.5 are correct in the sense that we can observe the discontinuity's location and shape. However, the sharpness still needs to be improved to be of the same standards as the benchmark results. Our ground truth material distribution only contains two unique values. A steep sigmoid function will help map the density values to the two extremes. In efforts to improve the sharpness of our final prediction, we implement an adaptive sigmoid function of the form $\sigma(n \cdot (x - a))$, in which we optimize the model parameters $n$ and $a$ using a gradient descent approach, hoping to achieve a better sharpness. The architecture used is shown in figure 3.2.

## 3.3. Experiment configuration and benchmark Results

For the implementation of our method, we have adapted the code provided in [21] following the paper [20]. As stated in the previous section, we consider a US NDT setting in which we want to identify a circular void inside a rectangular domain. We follow the same configuration as the 2D case in [20]. Displacement measurements are only available at sensors placed along the top boundary. The configuration for the experiments consists of the following: We use 54 sensor locations placed uniformly along the top boundary of the domain, and four sources also placed in the top boundary as shown in figure 3.3. In addition, the sine burst shown in equation 3.6, with $n_c = 2$ cycles and an amplitude of $10^{12}\text{kg(ms)}^{-2}$ models the source term. As a last remark, the density of the void-less material is $\rho_0 = 2700\text{kg/m}^3$

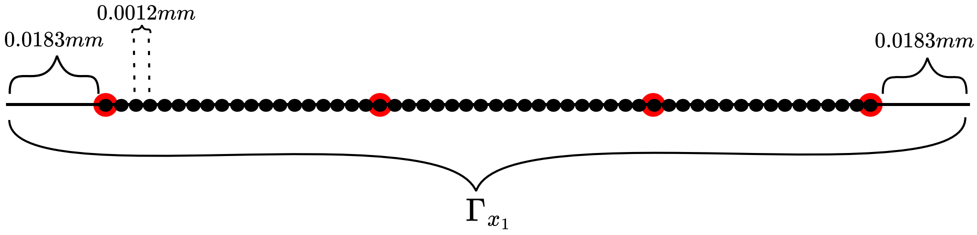


Figure 3.3.: Configuration of sensors (black circles) and sources (red circles) in the specimen used in our FWI experiments.

Moreover, we recall that in FWI we model the propagation of a wavefield in the specimen being studied at every iteration of the optimization algorithm. In this case the non-trainable forward solver is a second order finite difference solver, implemented in PyTorch and found in [20]. The rectangular domain used in the experiments under this section has a length of 0.1m and a width of 0.05m. We discretise the domain with a 2d grid of 254x126 cells, where the rows and columns at the boundaries are ghostcells. Furthermore, the cell sizes are obtained dividing the domain lengths by the number of cells in each dimension, the time step used in the forward simulation is derived from the CFL condition of the numerical scheme and is given by $\Delta t = 3 \times 10^{-8}$. Wavefield simulations are ran for 1200 time steps.

Figure 3.4 shows the wavefields obtained when solving the equation 3.1 for the ground truth material distribution.

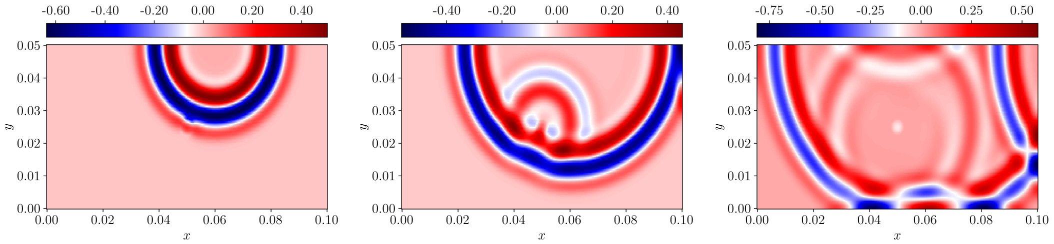
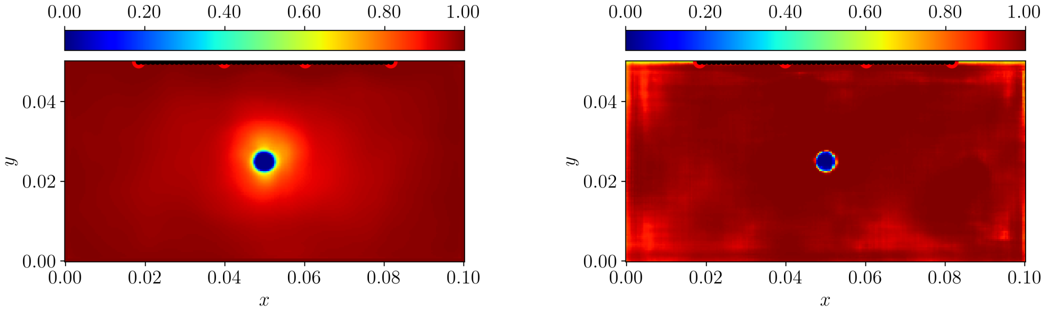


Figure 3.4.: Simulated wavefield in ground truth material distribution propagate at time steps 180, 270, 360. The emitting source is the third source starting from the left in figure 3.3.

Figure 3.5 shows the results obtained in [20]. These results are the benchmark to which we will compare the results obtained with our proposed method. In the paper, they solve the FWI problem starting with a PINN framework, changing specific components until they arrive to the state-of-the-art method in FWI - the adjoint state method. To get the results shown in 3.5 the material distribution is predicted using a convolutional neural network for figures 3.5a, 3.5b, and 3.5e. Case 3.5c employs a piece-wise polynomial ansatz $\hat{\gamma}(x) = \sum_{i=1}^{N_c} c_i N_i(x)$ for approximating the material distribution. The respective ansatz is defined by

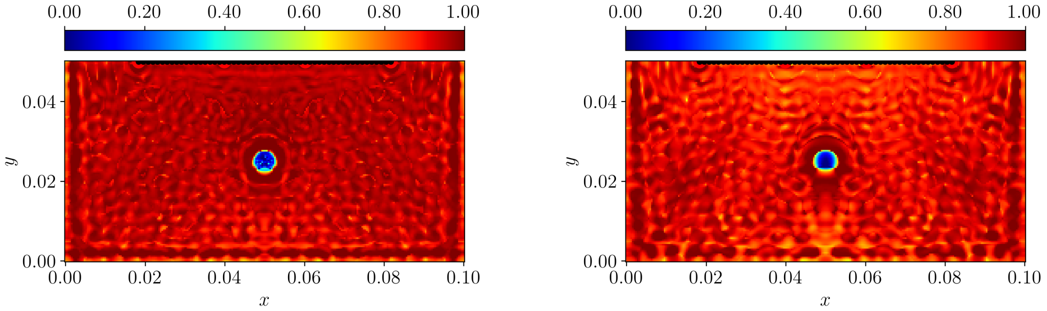
$$N_i(x) = \begin{cases} 1 & \text{if } x_i \leq x < x_{i+1}, \\ 0 & \text{otherwise.} \end{cases} \tag{3.8}$$

The coefficients are optimized using ADAM optimizer and the gradients are obtained through automatic differentiation implemented in PyTorch. The continuous adjoint method could also be employed for calculating the gradients of the loss function with respect to the coefficients of our constant ansatz. Furthermore, figure 3.5d shows the results obtained with the classic adjoint method. Here, the material distribution is predicted without any deep learning approach and gradients are computed with the continuous adjoint method. This result is the benchmark against what results obtained in figures 3.5a, 3.5b, 3.5c and 3.5e are compared with. Finally, the article proposes a new hybrid method to tackle the FWI task which combines the strengths of using a neural network ansatz to predict the material distribution and the strenghts of using the adjoint state method for computing the gradients instead of automatic differentiation. In this master thesis, we ideally want to implement a new version of the hybrid algorithm combining the strengths of the already implemented algorihtm in [20] and the SWIM weight sampling algorithm to speed up the inference when tackling the FWI task.



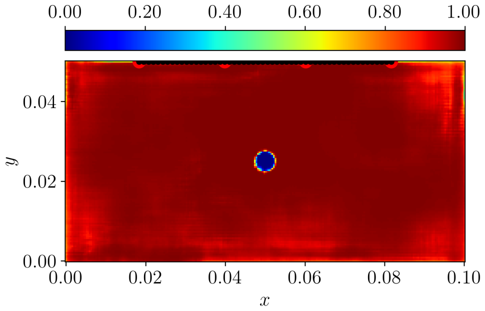
(a) PINN algorithm

(b) AD Neural network ansatz



(c) AD constant ansatz

(d) adjoint method



(e) hybrid method

Figure 3.5.: Results obtained for the FWI problem in [20].

How will our new proposed hybrid method help speed up solving the FWI problem? Our proposed hybrid method is based on a shallow feed-forward neural network, that exploits the computational advantages of the gradient computation using the adjoint state method to compute the derivatives of the loss function with respect to the material distribution's parameters, and a clever weight sampling technique for neural network weights that will allow us to only optimize for the weights of the output layer plus a bias term, with the aim of speeding up the inversion process. The aim of our method is to solve the FWI problem using a neural network with one hidden layer, in which the weights of the hidden layer are fixed before and during the optimization of equation 2.1. During the optimization loop, only the coefficients and bias term of the linear output layer will be optimized.

We conduct different numerical experiments to study how well we can solve the FWI problem with our proposed method. We conduct the following experiments:

- Train the output layer coefficients as well as the sigmoid parameters to solve the FWI problem for different initializations of our shallow network weights.
- Train for the steepness and shift coefficients of the sigmoid function after our output layer.

## 3.4. Numerical experiments for our proposed FWI method

### 3.4.1. Our approach for solving FWI task

Having explained the problem setup, configuration for our experiments, and the SWIM algorithm for sampling neural network weights, we now tackle the real inverse problem and evaluate how well our hybrid proposed method performs. Recall the SWIM algorithm used to sample weights for our feed-forward neural network. The forward pass of the hidden layer of the swim network can be represented as a matrix $A \in \mathbb{R}^{n \times k}$ where $n$ represents the number of input data points and $k$ represents the number of neurons in the hidden layer. In order to obtain the coefficients of the linear output layer, we can solve a linear system by solving a least squares problem, as the number of data points n is much larger than the number of neurons. We can augment matrix $A$ with an extra column of ones to account for the bias term in the solution that minimizes the least squares problem. Moreover, as we previously know that our specimen for the FWI problem is a rectangular domain with a circular void, we use the fact that we are considering a nearly intact object and this accounts to a domain where cells in our grid will have a value of $\gamma = 1$ nearly everywhere. With the intention of helping our method converge to the true solution, vector $\mathbf{b}$ will be initialized to an array where all entries are one. Finally, recall the steep and shifted sigmoid function used to make the output of the linear layer sharper. We need to apply the inverse of this function to our target to solve for the linear coefficients and bias term in the output layer of our neural network.

Given the linear system:

$$\mathbf{A}\boldsymbol{x}_{\text{coefficients}} = \sigma^{-1}(\mathbf{b}) \tag{3.9}$$

where $\mathbf{A} \in \mathbb{R}^{n \times k+1}$ with $n \gg k$ and $\mathbf{b} \in \mathbb{R}^n$, we solve for $\boldsymbol{x}_{coefficients}$ using the least squares method.

The least squares solution $\mathbf{x}$ minimizes the residual sum of squares:

$$\boldsymbol{x}_{\text{LS}} = \arg\min_{\mathbf{x}} \|\mathbf{A}\boldsymbol{x}_{coefficients} - \sigma^{-1}(\mathbf{b})\|_2^2 \tag{3.10}$$

Once solving the linear system, the solution $\boldsymbol{x} \in \mathbb{R}^{n+1}$ contains the coefficients and bias term of the output linear layer of our neural network. Having solved the least squares problem, we initialize the output linear layer of our shallow feed-forward neural network with the solution obtained with equation 3.10 and we try to solve the FWI problem.

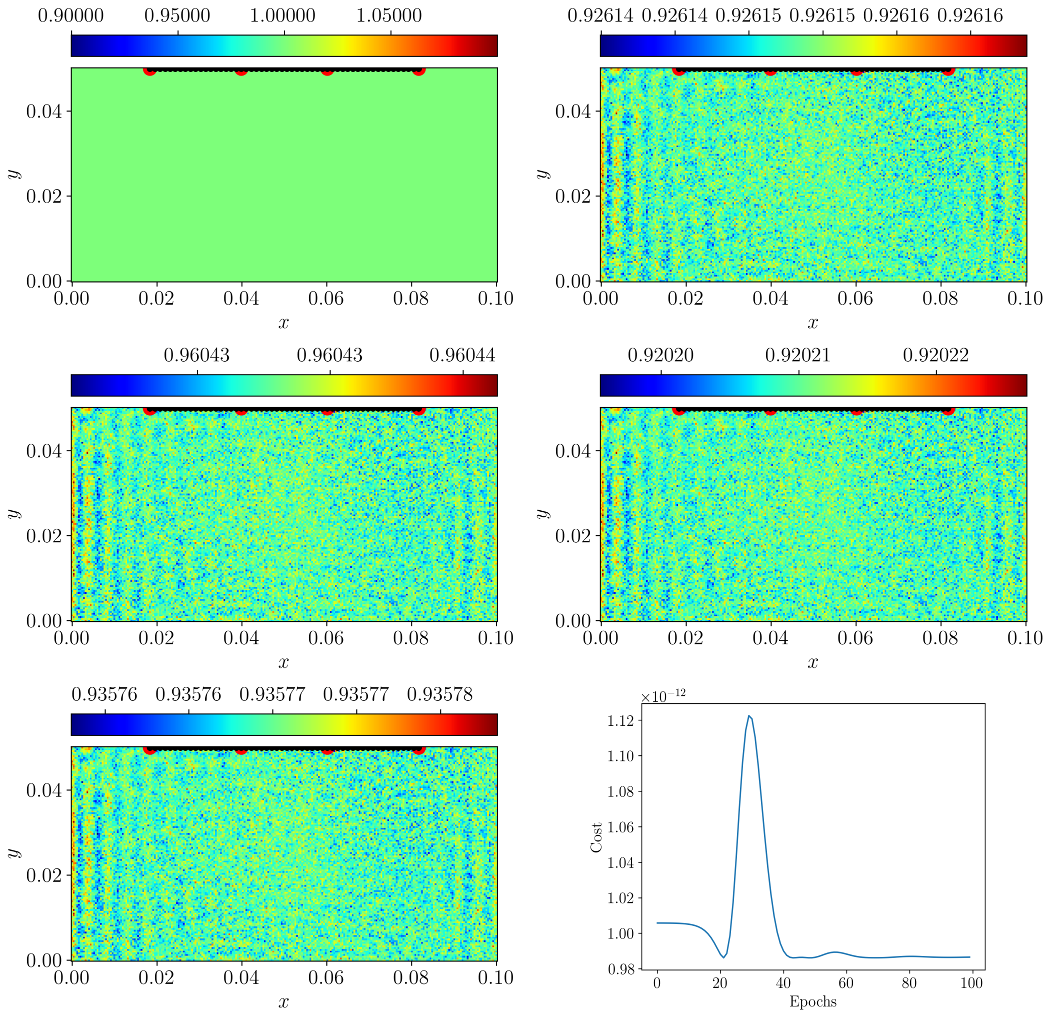


Figure 3.6.: Plots show the evolution of the predicted material distribution over 100 training epochs. We show the results after epochs 0 (top left image), 20 (top right), 40, 60 and 100 (bottom left) and the cost during training.

Figure 3.6 shows how the predicted material distribution evolves during the training epochs, as well as the cost evolution. We can observe that the cost reaches a plateau and does not improve further, and that the result obtained after training is disappointing. The magnitude of the cost is very small, of the order of $10^{-12}$. This has an explanation: The cost is calculated with equation 2.1. We integrate in time using the trapezoidal rule, which uses the timestep as a multiplicative constant. The timestep used in the simulations is of the order of $10^{-8}$. We integrate in space by multiplying by the grid cells' area, which is of the order of $10^{-4}$. Multiplying these small quantities makes the evaluation of equation 2.1 yield very small results.

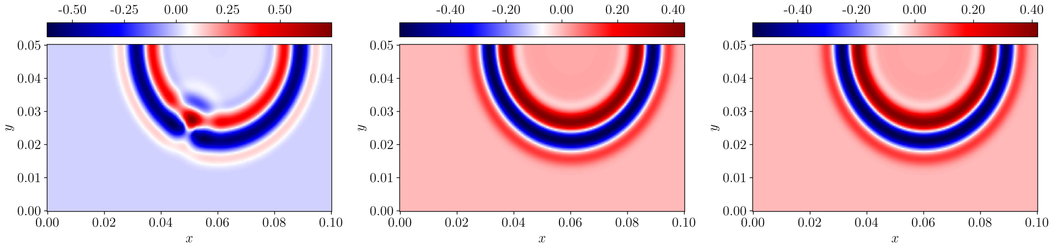


Figure 3.7.: Plots show simulated wavefields using the ground truth gamma distribution, the initialized gamma distribution and the converged gamma at timestep 220.

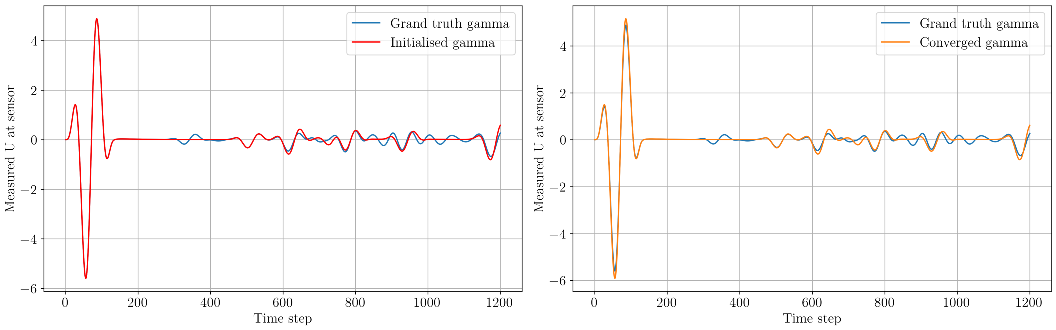


Figure 3.8.: Diagrams show the time-series measurements at sensors in the top boundary at x position 220.

Figures 3.7 shows the wavefields simulated in the ground truth material distribution and in the initialized and converged material distributions, before and after the optimization procedure respectively. Moreover, figure 3.8 shows the time-series values measured in the sensor located at grid point in the top boundary with x-position 220 for both the initialized gamma distribution and the converged gamma distribution. This initialization does not show much improvement. Table 3.1 quantifies the misfits between ground truth gamma, the initialized gamma, and the converged gamma, for all sensor values at all time-steps.

### 3.4.2. FWI results with different weights and biases initializations

#### Initializing weights ans bias terms to approximate a large void in the domain

Our previous initialization was not adequate. We did not see much improvement in the cost function. In our next experiment we aim to overcome the convergence problems of our method by fitting a big void in the domain. In previous experiments, the coefficients in the output linear layer were obtained by solving a linear system. The right hand side of this linear system was a vector full of ones, obtained from a void-less domain, reshaped to satisfy the shape requirements to solve for the vector of coefficients. Now, instead of solving the linear coefficients for a domain full of ones, we solve for the linear coefficients for a domain with a big circular void, much larger than the true circular void we aim to predict. After 100 epochs of training, we obtain the results shown in figure 3.9.

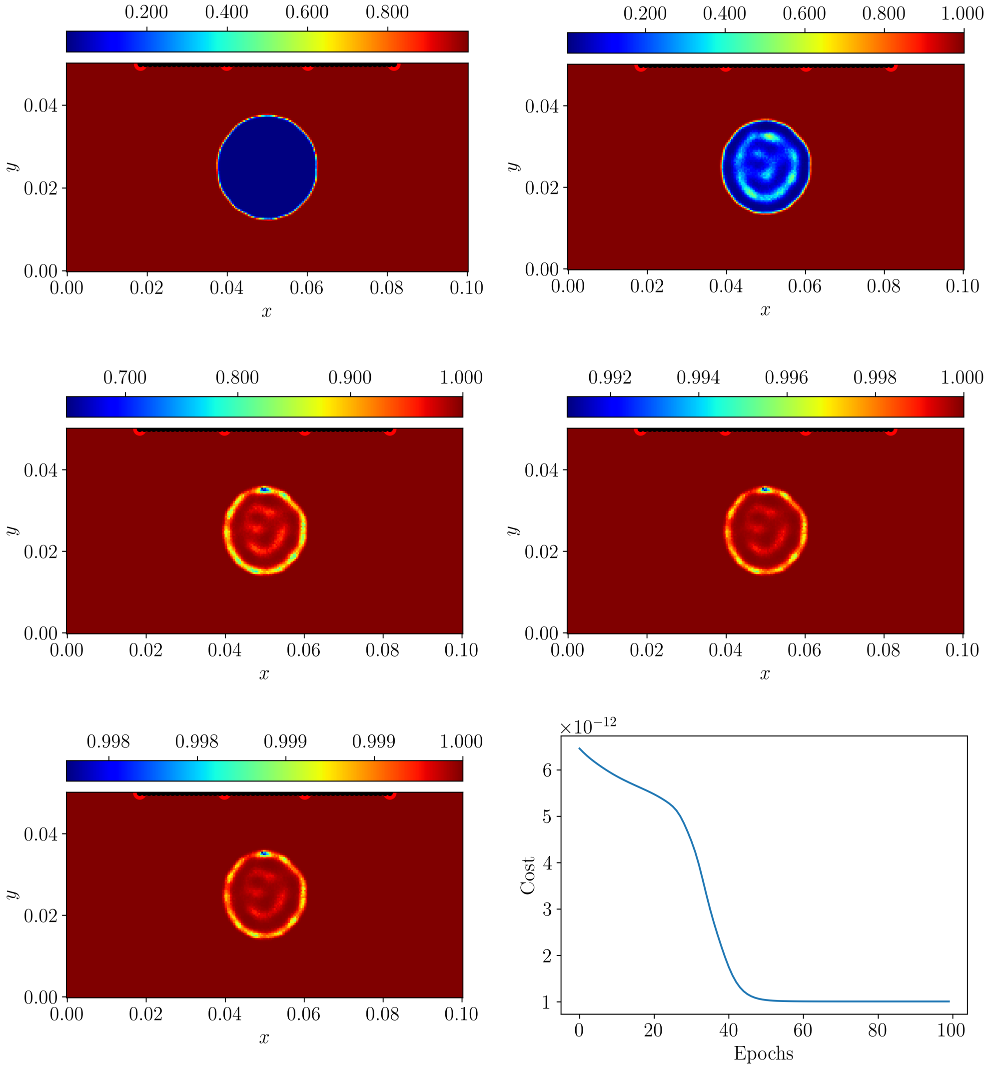


Figure 3.9.: Plots show the evolution of the predicted material distribution over 100 training epochs. Plots are shown for epochs 0 (top left image), 25 (top right), 50, 75 and 100 (bottom left). We also show the cost evolution during training.

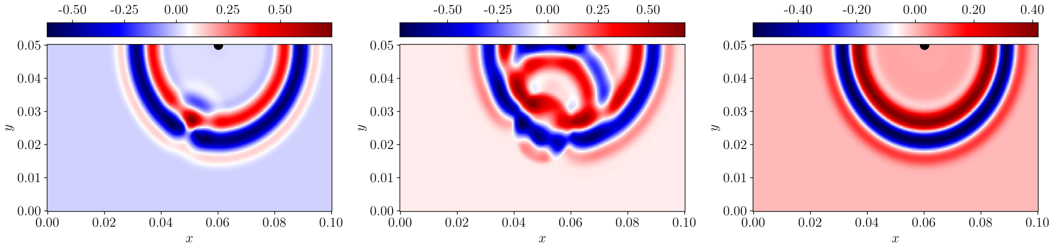


Figure 3.10.: Diagrams show a wavefield at timestep 220 for a wave simulated using (i) the ground truth gamma (left image), (ii) the initialized gamma (middle image), and (iii) the converged gamma (right image)

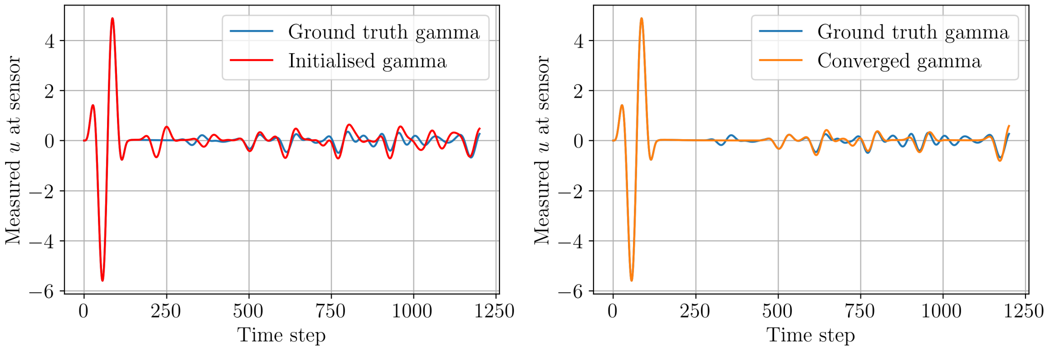


Figure 3.11.: Plots show the time-series of the displacements measured by the sensor at the top boundary of the specimen, at x position 152. The Left image shows the time-series if we simulate the wave propagation with the material distribution shown in the top left image of figure 3.9. The right plot shows the time-series if the wave was propagated in the converged material distribution, shown in the bottom left image of figure 3.9

Figures 3.9 , 3.10, and 3.11 all suggest that our optimization was correctly performed, we see the cost decrease and the initial big void in the domain fills up. Moreover, the trajectories shown in 3.11 back up and validate the decreasing cost. The trajectory obtained from simulating a propagating wavefield in the converged material distribution is noticeably closer to the ideal trajectory than the one obtained from simulating the wave propagation in the material distribution with the big void. However, we reach a plateau in the training cost and we are unable to converge to the ground truth material distribution.

### Initializing weights and bias terms to approximate a shifted version of the ground truth function

Moreover, the question we asked ourselves to design the next experiment is: How can we further help our algorithm converge to the ground truth gamma? Having the same idea in mind as in the last experiment, we want to solve the linear coefficients of the output layer to fit a very similar gamma function as the ground truth, even more similar than the big circular void of the experiment above. This time, we will fit a slightly shifted circular void, trying to initialize the weights and biases of our network to something very close to the ground truth, hopefully helping the training converge to the true solution and see if gradient descent works.

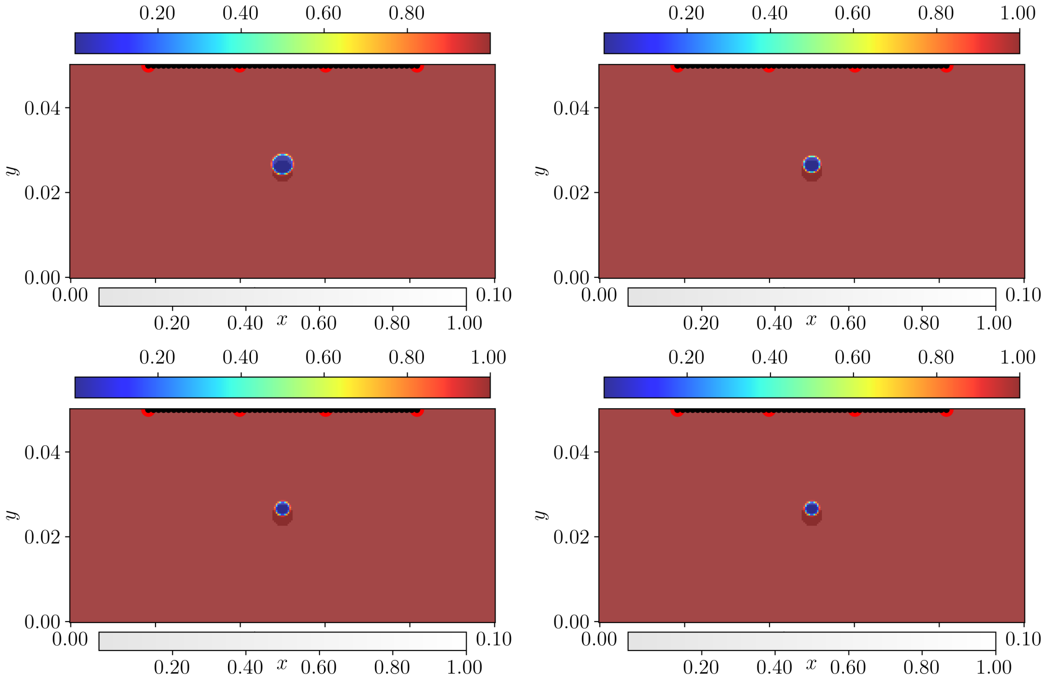


Figure 3.12.: Plots show the evolution of the predicted material distribution over 100 training epochs. Plots are shown for epochs 0 (top left image), 10 (top right), 40 (bottom left), and 100 (bottom right).

Unfortunately, we did not get the results we expected when performing our experiments. Although the misfits shown in table 3.1 and the trajectories plotted in 3.14 suggest that we improved the prediction, we did not converge to the ground truth gamma distribution.

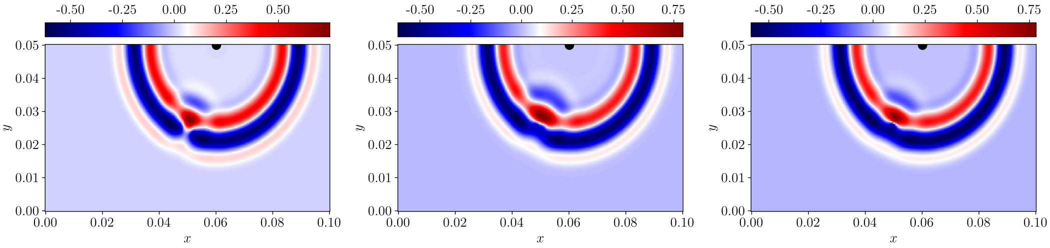


Figure 3.13.: Plots show a wavefield at timestep 220 for a wave simulated using (i) the ground truth gamma (left image), (ii) the initialized gamma (middle image), and (iii) the converged gamma (right image).

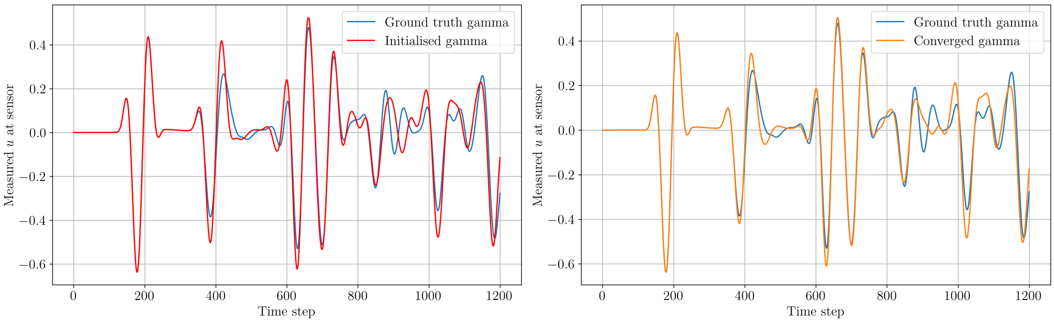


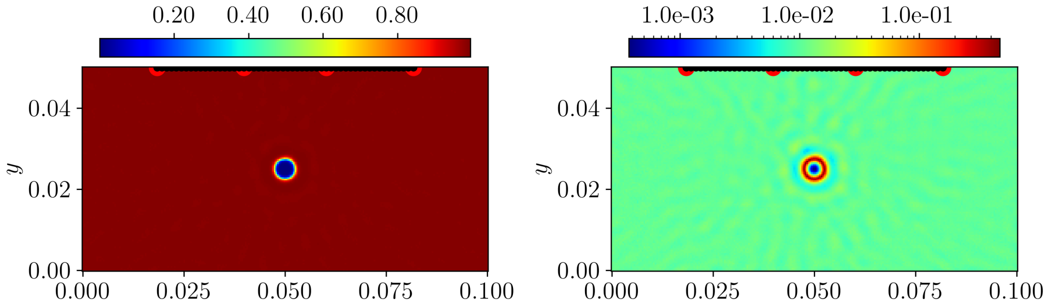
Figure 3.14.: Plots show the time-series of the displacements measured by the sensor at the top boundary of the specimen, at x position 152. The Left image shows the time-series if we simulate the wave propagation with the material distribution shown in the top left image of figure 3.12. The right plot shows the time-series if the wave was propagated in the converged material distribution, shown in the bottom left image of figure 3.12

Table 3.1.: Misfit values for different experiments

| Experiment | Initial Misfit | Converged Misfit |
|---|---|---|
| Domain of ones (subsection 3.4.1) | 1333.11 | 1226.04 |
| Double Right Shift (appendix A) | 1574.28 | 1185.64 |
| Right Shift (appendix A) | 601.23 | 600.88 |
| Upwards Shift (subsection 3.4.2) | 1033.92 | 744.12 |
| Large Void (subsection 3.4.2) | 10057.38 | 1226.08 |

### 3.4.3. Training sigmoid parameters in FWI setting

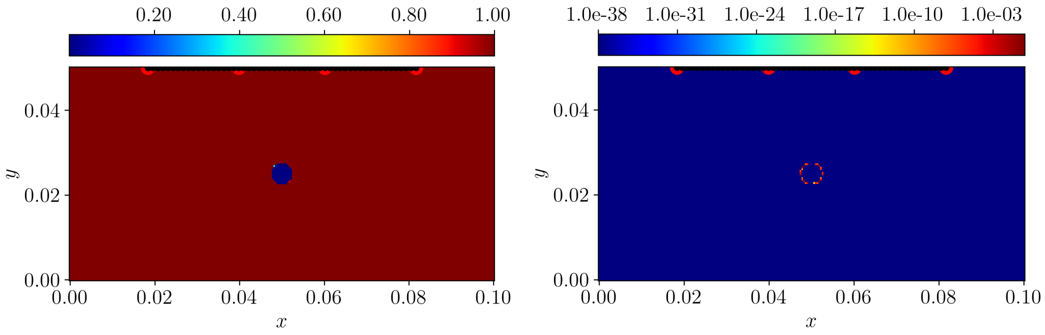
For this experiment, the objective is to check whether we can fit the ground truth gamma exactly with our SWIM network and gradient descent. In previous experiments, we have gradually made the problem simpler and simpler by changing the initialization of our weights. This is the last experiment with this particular setting. Here, we will fit the ground truth material distribution function with our SWIM network. We then take all weights and bias terms of the SWIM network, and initialize our network with these weights and fix them. As observed in figure 2.4, the SWIM network does not produce a binary output, therefore we need a very steep sigmoid function to map the values to its extremes. We use a sharp and shifted sigmoid function, so that the output of our network is mapped as close as possible to 0 or 1. The form of this sigmoid is $\sigma(n \cdot (x + a))$ where $n$ is the sharpness coefficient indicating the steepness of the function and $a$ is the shift of the sigmoid function. The steepness and shift coefficients will be the only trainable parameters in this experiment, the weights and bias terms used during the initialization already produce a good fit of the ground truth gamma.



(a) Gamma prediction

(b) Error in the prediction

Figure 3.15.: Plots show the predicted material distribution (gamma) and the error in the prediction obtained using an adaptive sigmoid function with trainable shift and sharpness coefficient. $n$ was initialized to 10 and $a$ to 0.5 before training started.



(a) Gamma prediction

(b) Error in the prediction

Figure 3.16.: Plots show the predicted material distribution (gamma) and the error in the prediction obtained using an adaptive sigmoid function with trainable shift and sharpness coefficient. $n$ was initialized to 336.774 and $a$ to 0.547 before training started.

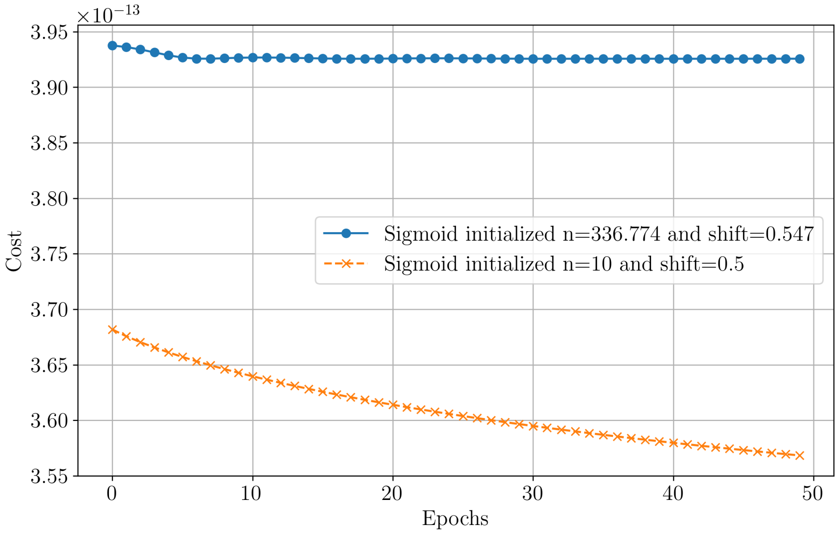


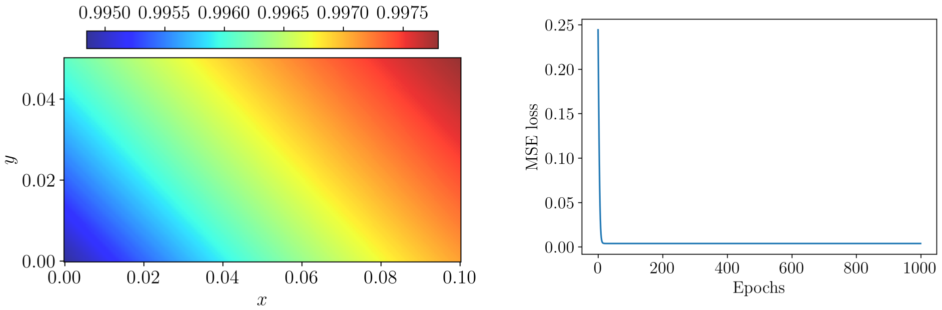
Figure 3.17.: Plot shows the evolution of cost during training of parameters in adaptive sigmoid function

Figures 3.15, and 3.16 show the result of the approximation for different initializations of the coefficients in our sigmoid function. Looking at figures 3.15, 3.16, and 3.17, shows that our gradient descent optimizer faces challenges trying to find the shift and sharpness coefficient needed to make the fitting of the ground truth gamma perfect. Figure 3.17 shows the evolution of the costs over the epochs, and indicates that the sigmoid function initialized to $n = 10$ yields a better cost after 50 epochs of training. However, the gamma prediction and error show the opposite. This showcases the challenge faced when training the sharpness and shift parameter of our activation function to make the fitted ground truth look sharper. We now start to think that our proposed method for solving the FWI challenge will not work. In experiments below we try to confirm this hypothesis and we tackle a simpler problem than the FWI task.

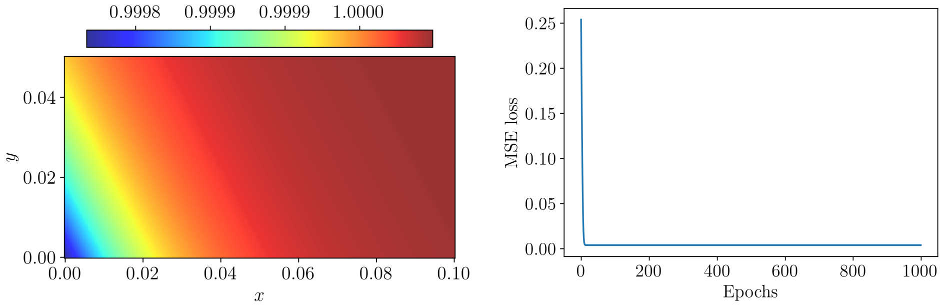
For the experiments above, the cost decreased during training between two and three orders of magnitude and then reached a plateau which did not improve with increasing number of epochs. In order to validate our approach, we plotted the time-series values of one sensor in the top boundary of our domain for three material distributions for each of our experiments. These three material distributions are: the ground truth, the material distribution after the first forward pass of the model, and the material distribution obtained after training. For example, regarding the big void experiment, figure 3.11 shows the respective trajectories for the three distributions. These trajectories represent the displacement values captured at one sensor for all timesteps when a wave is propagated in the respective material distribution. We can observe that the misfit between the trajectories in the two plots decreases from the trajectory in the initialized material distribution, to the trajectory in the converged material distribution. The decrease in misfit agrees, and drives the decreasing training cost. However, the decrease was not enough and our model, alongside with the respective initialization, could not solve the FWI problem in regard. Table 3.1 contains all the misfits with respect to the ground truth gamma trajectory, for the initialized distribution and the converged distribution. We can see that all converged misfits are smaller than the initialized misfits, showing the training was done correctly, but the converged solution was not the optimal.

### 3.4.4. Supervised learning experiments and gradient descent

Results of experiments above suggest that our inverse problem is difficult to solve with gradient descent and our shallow network. The plotted trajectories shown for each initialization tested serve as validation for our experiments. For all our experiments above the cost decreased during training until a plateau was reached, however the converged solution is not the correct one. We now conduct experiments to see if our ground truth gamma function can be approximated using our neural network in a supervised learning framework. The supervised learning problem should be easier to solve than the FWI problem. If the supervised learning problem cannot be solved with our network architecture, we will not be able to solve the inverse problem using the same architecture as this constitutes a harder problem.



(a) Supervised learning task using the proposed network architecture for the FWI problem.



(b) Supervised learning task for an MLP with three hidden layers.

Figure 3.18.: Plots show results obtained for the supervised learning task for (a) our proposed network architecture and (b) an MLP with three hidden layers. The images on the left hand side show the prediction after training.

On the one hand, Figure 3.18 suggests that our shallow network is not able to solve the supervised learning task for the given discontinuous material distribution. We observe how the MSE loss reaches a plateau, and the loss will not decrease further. We have attempted to solve the supervised problem with two network architectures. Figure 3.18a shows the result when exploiting the capabilities of a FFN with one hidden layer, whereas 3.18b shows the result of the fitting when using a FFN with three hidden layers. We can say there is a slight increase in the quality of the fitting, but the results suggest a shallow feed-forward network cannot capture the discontinuity of the ground truth gamma studied in our FWI problem, and as a result, our method fails to solve the FWI problem under consideration. On the other hand, figure 3.19 shows that the CNN employed to solve the FWI problem for figure 3.5 is able to solve the supervised learning problem up to a very high accuracy, contrary to the feed-forward networks we tested. The good results yielded by the CNN in the supervised learning task makes attempting to solve the FWI problem with this same architecture a promising experiment, as it turns out by the results presented in [20] which employ the same CNN to solve the FWI problem.

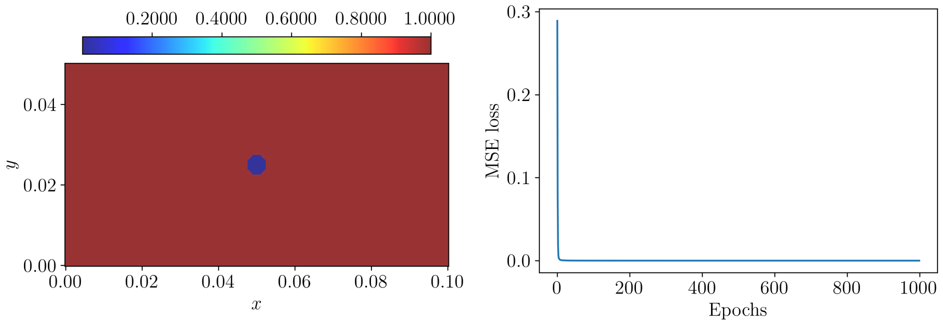


Figure 3.19.: Plots show the result and training for the supervised learning task using CNN architecture.

## 3.5. Smooth function experiments

Experiments from the supervised learning framework suggest that our discontinuous function is hard to approximate with our shallow network using gradient descent. We now change the ground truth gamma function to a smooth function to see if this overcomes the problem, as smooth functions are easier to approximate with neural networks. We will run experiments for different settings. One setting will consider a ground truth material distribution which is the sigmoid of a Gaussian function. The second setting will explore having a ground truth material distribution which is a sigmoid of a two-dimensional sine function. We include the sigmoid function in order to squeeze the output between 0 and 1, agreeing with the parametrization of voids in the problem description, described by equation 3.3.

**Smooth function 1:**

We define a Gaussian function over the spatial domain $[0, L_x] \times [0, L_y]$. The function is centered at $(\mu_x, \mu_y)$ and is expressed as follows:

$$G(x, y) = \exp\left(-\frac{(x-\mu_x)^2}{2\sigma_x^2} - \frac{(y-\mu_y)^2}{2\sigma_y^2}\right), \tag{3.11}$$

with

$$\mu_x = \frac{L_x}{2}, \quad \mu_y = \frac{L_y}{2},$$

and $\sigma_x$ and $\sigma_y$ are the standard deviations in the $x$ and $y$ directions, respectively. Both standard deviations are set to 0.01. To obtain the smooth function for our experiments, $\tilde{G}(x, y)$, we first normalize and invert the Gaussian function:

$$\tilde{G}(x, y) = s \cdot (1 - G(x, y)) + \epsilon, \tag{3.12}$$

where $s$ is a scaling factor and $\epsilon$ is a small minimum value to ensure the function remains bounded away from zero, identical to the lower bound of equation 3.3. Finally, we apply a sigmoid transformation to $\tilde{G}(x, y)$:

$$\text{Sigmoid}(\tilde{G}(x, y)) = \frac{1}{1 + \exp(-\tilde{G}(x, y))}. \tag{3.13}$$

**Smooth function 2:**

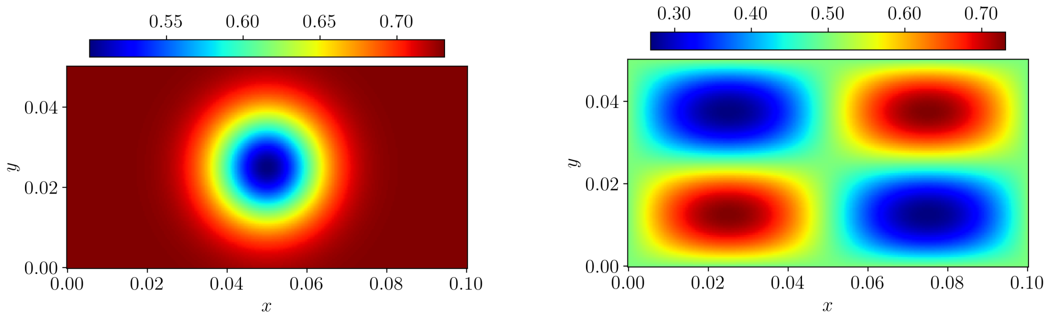
We define a sinusoidal function $S(x, y)$ over the spatial domain $[0, L_x] \times [0, L_y]$ as:

$$S(x, y) = \sin\left(\frac{2\pi x}{L_x}\right) \cdot \sin\left(\frac{2\pi y}{L_y}\right), \tag{3.14}$$

where $L_x$ and $L_y$ are the lengths of the domain in the $x$ and $y$ directions, respectively.

Next, we apply the sigmoid function to the sinusoidal function $S(x, y)$ to obtain a smooth output:

$$\text{Sigmoid}(S(x, y)) = \frac{1}{1 + \exp(-S(x, y))}. \tag{3.15}$$



(a) Sigmoid of Gaussian function

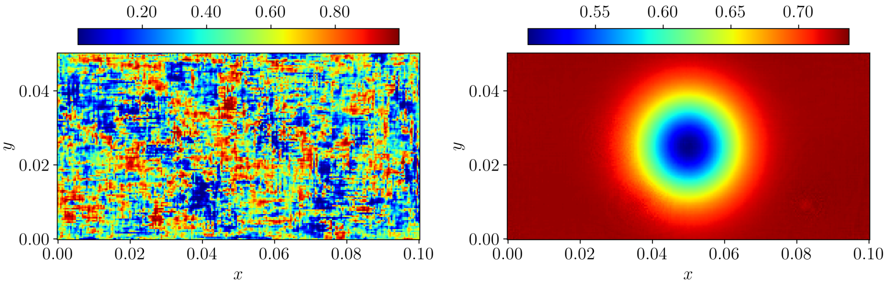
(b) Sigmoid of Sine function

Figure 3.20.: Plots show the two ground truth smooth functions. Figure (a) shows the sigmoid of the inverted Gaussian function 3.13, (b) shows the sigmoid of a sine function 3.15. We will be conducting experiments with these functions as ground truths.

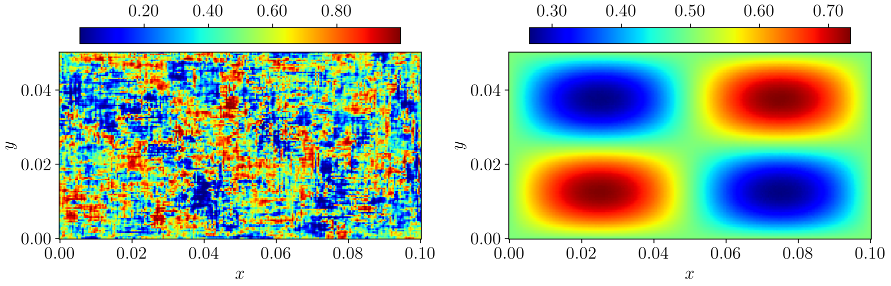
We will revisit our approach to solve the FWI problem for the two material distributions displayed in figure 3.20. We will first study the supervised learning task with three different neural network architectures: (i) CNN, (ii) feed-forward neural network with one hidden layer, (iii) feed-forward neural network with three hidden layers. The result of the supervised learning task will again give us an idea about the potential of each respective network when attempting to solve the FWI problem. In addition, we will test two different initializations when attempting to solve the inverse problem, namely, Xavier-Glorot initialization, and initializing the weights of the network with the weights of the same network trained in a supervised learning setting to approximate a voidless domain. We conduct this experiment to see if initializing the weights after fitting a domain of ones is problematic, as this yielded strange results when attempting to solve the FWI problem with the proposed method of this master´s thesis.

### 3.5.1. Supervised learning experiments

For the supervised learning problems, we initialize the network using Xavier-Glorot initialization. The dimensions of the rectangular domain modeling the specimen are the same as in the FWI problem. Moreover, we employ a constant learning rate of $lr = 0.001$ and choose to use ADAM as our optimizer. The loss function to optimize is the Mean Squared Error loss due to the continuous nature of the function we want to fit with our models. For the supervised learning experiments with CNNs we train for 5000 epochs as the result got noticeably better when training for longer. Figure 3.21 shows the result of the supervised learning task when employing a CNN.



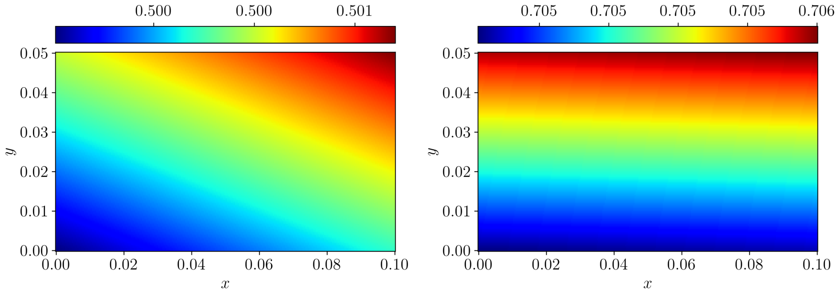
(a) Sigmoid of Gaussian function



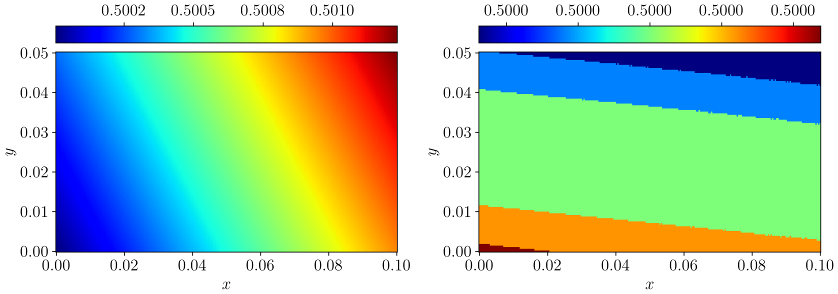
(b) Sigmoid of sine function

Figure 3.21.: Plots show the results for the supervised learning task for (a) sigmoid of Gaussian function and (b) sigmoid of sine function. We trained the CNN for 5000 epochs with Xavier-Glorot initialization. Images on the left are the results of the first forward pass. Images on the right are the converged results.

As a previous, cautious step before attempting to solve the FWI problem with our shallow network and the clever weight sampling algorithm, we test the ability of our network to solve the supervised learning problem. Our network has the same architecture as described in figure 3.2. Regarding the training, we also use a constant learning rate of $lr = 0.001$. We have tested bigger and smaller learning rates but these do not yield better results. As for the supervised learning experiment using the CNN for our smooth functions, we also use ADAM optimizer and our loss function is also the MSE loss. Finally, we include a regular sigmoid function with no shift or steepness parameters, as our target function is continuous. Figure 3.22 shows the results obtained for the described training configuration and architecture.



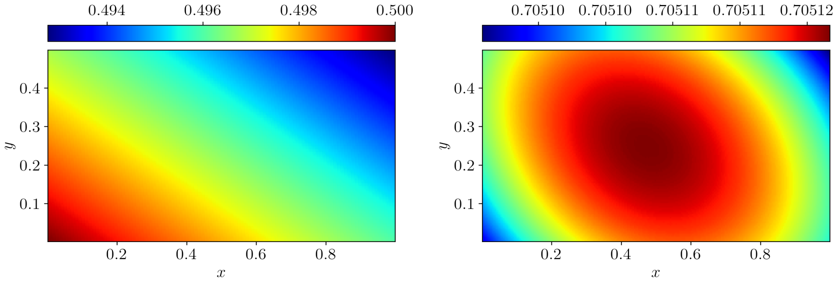
(a) Fitting sigmoid of Gaussian function with feed-forward one hidden layer network.



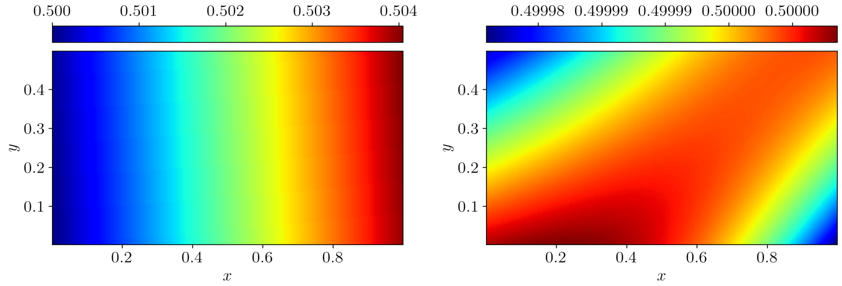
(b) Fitting sigmoid of sine function with feed-forward one hidden layer network.

Figure 3.22.: Plots show the results for the supervised learning task for (a) sigmoid of Gaussian function and (b) sigmoid of Sine function using our network architecure (one hidden layer), Trained for 1000 epochs with Xavier-Glorot initialization. Images on the left hand side are the result after the first forward pass whereas images to the right are the result of the converged predictions.

Again, our shallow network cannot capture the properties of the target function. What happens when we increase the number of layers? We now perform the experiments yielding the results shown in figure 3.22 but this time using a feed-forward neural network with three hidden layers. We also use $\tanh$ activation after the hidden layers, and a sigmoid function after the linear output layer. Results for these experiments are shown in figure 3.23. We observe this network approximates the functions better yet still not to the desired accuracy. It is not worth testing the FWI problem using these architectures and these target functions. For the supervised learning experiments using multiple layers, we trained for 1000 epochs as the cost did not decrease any further for our given setting.



(a) Fitting sigmoid of Gaussian function.

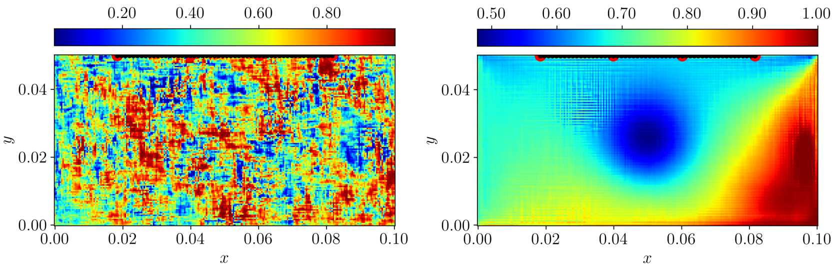


(b) Fitting sigmoid of sine function.

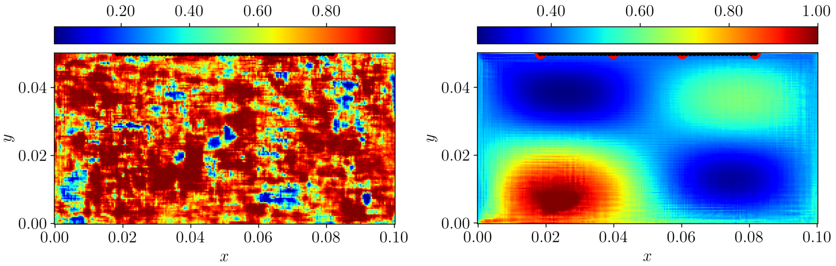
Figure 3.23.: Plots show the results for the supervised learning task for (a) sigmoid of Gaussian function and (b) sigmoid of sine function using MLP with three hidden layers, Trained for 1000 epochs with Xavier-Glorot initialization. Images on the left show the result after the first forward pass. Images on the right show the final result.

### 3.5.2. FWI with CNNs and smooth functions

Results obtained for the supervised learning experiments using the CNN architecture highlight the potential of this architecture for solving the FWI problem for the smooth functions setup. Figure 3.24 shows the results obtained. We initialize CNN with Xavie-Glorot initialization. The plots on the left of figure 3.24 show the result after the first forward pass of the network, whereas the images on the right hand side show the converged results. We notice the results are not as sharp and accurate as the ones displayed in figure 3.5. The reason for this could be that there is not an actual void inside the domain. Figure 3.25 shows how the wavefields propagate in the whole domain, with no reflections in the interior of the domain. The smooth nature of the target function implies reflective effects only occur at the boundaries of the domain. In the presence of a void, we would obtain reflective phenomena at the boundaries of this void and thus have a bigger effect in the recorded measurements at the collocation points located at the top boundary. This is because a reflected wave from the middle of the domain would arrive at an earlier timestep than a reflected wave from the bottom boundary.



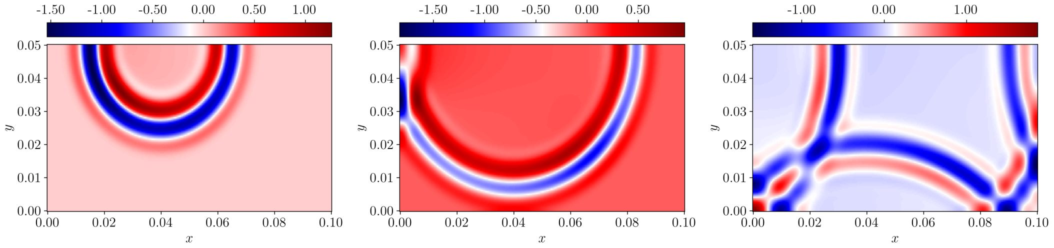
(a) FWI for sigmoid of gaussian function.



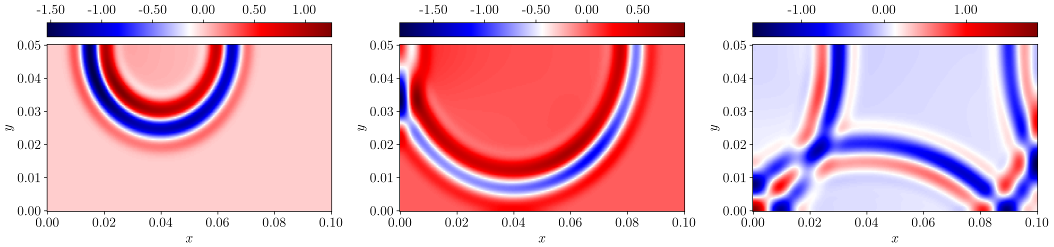
(b) FWI for sigmoid of sine.

Figure 3.24.: Plots show the results for the FWI task for (a) Gaussian function and (b) sigmoid of sine function for 2000 training epochs using a CNN. Images on the left show the first forward pass. Images on the right show converged result.

Moreover, we see that the wavefields obtained when solving the scalar wave equation in our smooth functions in figure 3.25 are identical to the human eye. The rather small domain and the rather small range of values in the images of our functions (around [0.3-0.7]) make the wave propagate at similar speeds. In addition, the short time for which the wavefield propagates does not allow the wavefield speed propagation values to diverge significantly. All of these arguments, in combination to a lack of void, make both wavefields appear very similar when propagating in our rectangular domain.



(a) Wavefields propagating in Gaussian function



(b) Wavefields propagating in sine function

Figure 3.25.: Plots show wavefields propagating in studied smooth functions for timesteps 200, 300, and 650.

## 3.6. Extended domain experiments

Since the beginning of the experiments in this master's thesis, we have tried to make experiments simpler and simpler to help the studied neural network architectures converge to a good solution. We first tried to solve the FWI task which proved to be impossible with our shallow network. Moreover, the supervised learning experiments also failed with the same shallow network, leading us to study smooth functions, striving to make the problem simpler again. We have tried to fit smooth functions with the CNN architecture used in [20], our network architecture (feed-forward network with one hidden layer), and a feed-forward neural network with four hidden layers. We designed the set of smooth function experiments to make it easier for our network to solve the FWI task. To our surprise, only the CNN was able to fit both smooth functions perfectly, and hence attempting to solve the FWI problem for these functions with our shallow network was not worth the effort. The smooth functions were represented and defined in the domain used in the original FWI problem as it can be seen in figure 3.20. We have seen however, that making the domain bigger allows the MLP with four hidden layers to fit the smooth function. The domain used in this case is ten times bigger than the domain studied in the sections above. In particular, we look at a domain $[0, 1.0] \times [0, 0.5]$. In this setting, we first explore the potential of our architectures when solving the supervised learning task. We expect the CNN to solve the supervised learning task with no difficulty as this experiment should be easier to solve than the previous ones, which were already successful. These experiments should be easier becase given the larger domain, the variations in the target functions are less abrupt. In this setting, we hope the MLP architecture is able to solve the supervised learning task to give as a chance of attempting to solve the FWI problem. The settings used in experiments with regular domain and extended domain are summarised in table 3.2.

Table 3.2.: We use the following parameters in the experiments. The first row of parameters accounts for sections 3.3, 3.4.1 and 3.5. The second row accounts for experiments in subsection 3.6.

| **Lx** | **Ly** | **Nx** | **Ny** | **dt** [s] | **N** | **gamma0** | **rho** $[kgm^{-1}]$ | **c** $[ms^{-1}]$ | **N. Sensors** | **sensorStart** [m] | **sensorEnd** [m] | **N. Sources** |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0.1 | 0.05 | 251 | 123 | 3e-08 | 1200 | 1e-5 | 2700.0 | 6000.0 | 54 | 0.018 | 0.082 | 4 |
| 1.0 | 0.5 | 251 | 123 | 3e-08 | 1200 | 1e-5 | 2700.0 | 45000.0 | 100 | 0.1 | 0.9 | 4 |

As we expected, the CNN is able to perfectly fit the smooth functions in the extended domain, corresponding to the settings of the second row in table 3.2. The results of the supervised learning task are displayed in figure 3.28. To our relief, figure 3.29 shows very promising results because the MLP successfully fits the desired functions. Given the accurate approximation, we will study the FWI problem using this architecture and both smooth functions represented in the extended domain. To attempt to solve the FWI problem, we first need to generate a dataset to be used as reference values in the misfit function 2.2. Given the slightly bigger domain, we have tried to keep things as constant as possible. We have kept the same values for generating wavefields using the forward solver, except for the cell sizes in the discretization (larger domain but same number of grid points), velocity $c$, and the frequency of the sine burst used as the forcing terms. We have set the velocity to $c = 45000\ ms^{-1}$ in order to obtain a wavefield that propagates in all of the domain, in the same number of timesteps as before (1200). Finally, we have also changed the frequency of the sine burst used as forcing term. Previous experiments used a frequency of 50000 Hz. For the experiments below, we will use a frequency of 33333 Hz. This value was chosen graphically from the wavefield snapshots. Figure 3.26 shows three snapshots at three different timesteps of a wave propagated in a specimen with a material distribution modeled by the Gaussian of the two-dimensional sine wave in the extended domain. Additionally, figure 3.27 shows the wavefield propagating in the Gaussian material distribution. The main difference to observe in figure 3.26 and 3.27 compared to figure 3.4 is the lack of a void in the domain, leading to reflections in areas closer to the sensors in the top boundary. In the case of the smooth functions, the density is never zero, and hence the wavefield propagates everywhere in the interior of the domain. Reflection only occurs at the boundaries, whereas in figure 3.4 we can observe the void reflecting waves in the middle of the domain.

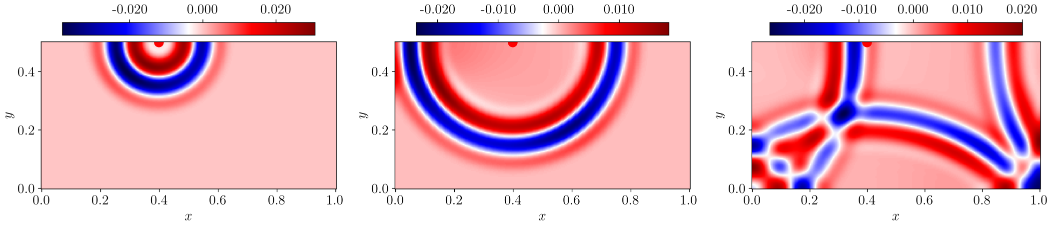


Figure 3.26.: Plots show simulated wavefields in sigmoid of sine material distribution in extended domain at timesteps 200, 350 and 650 (from left to right)

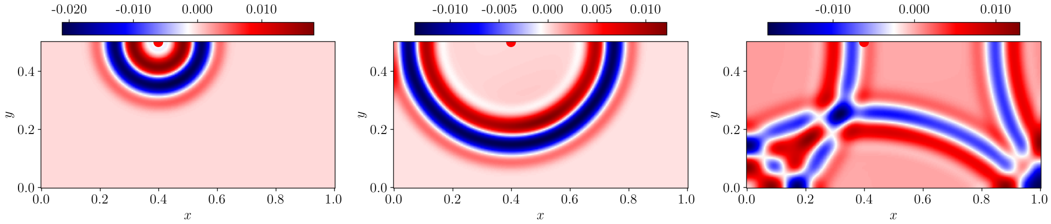


Figure 3.27.: Plots show simulated wavefields in sigmoid of Gaussian material distribution in extended domain at timesteps 200, 350 and 650 (from left to right)

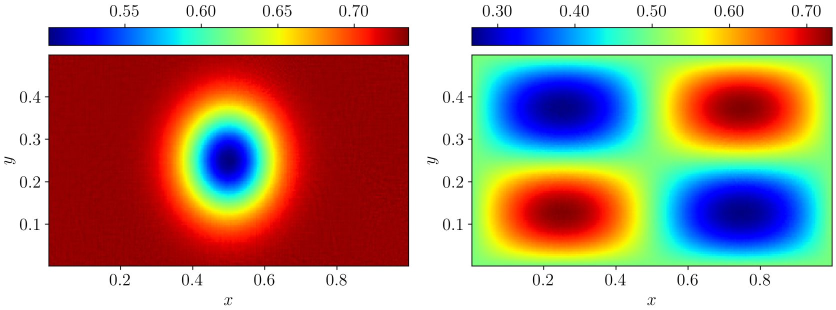
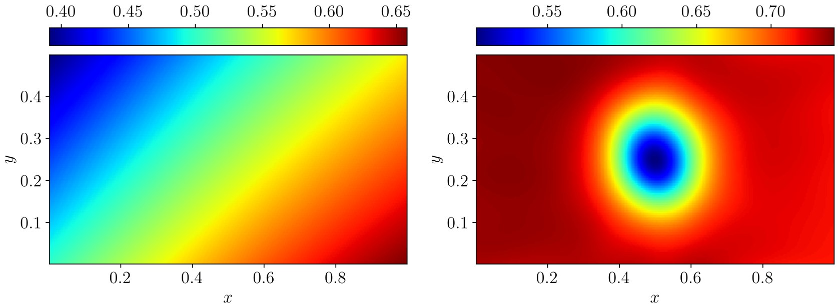
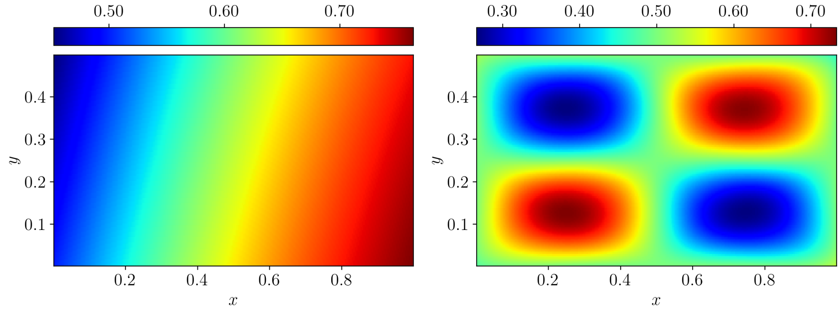


Figure 3.28.: Plots show results for the supervised learning task for sigmoid of Gaussian function (left) and sigmoid of sine function (right) using CNN. We trained the CNN for 3000 epochs using Xavier-Glorot initialization.



(a) Fitting sigmoid of Gaussian function.

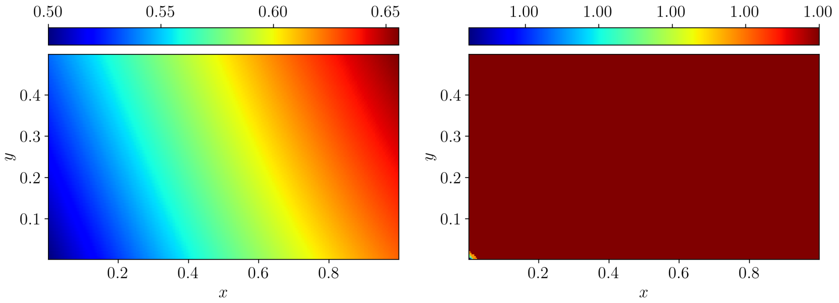


(b) Fitting sigmoid of sine function.

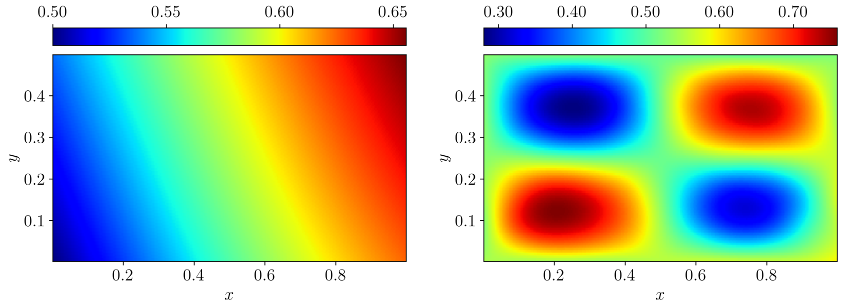
Figure 3.29.: Plots show results for the supervised learning task for (a) sigmoid of Gaussian function and (b) sigmoid of sine function using feed-forward neural network with four hidden layers and 350 neurons per layer. We trained the network for 2500 epochs with Xavier-Glorot initialization. The images on the left show the result after the first forward pass of the network. The images on the right show the converged result.

As a last remark before we attempt to solve the FWI problem, it might stand out to the reader that we have not included information about the error in the fitting of the functions above. We have an explanation for this: generally, when performing supervised learning tasks, the data is split into training, validation, and test sets. In our case, the input data is the whole domain and we do not perform such split. In our supervised learning framework, we want to know if our network architecture can be used to approximate a function, rather than making a future prediction. We thereby do not make any splits in our data, and use the whole 2-d domain as input to our network. To provide error of the supervised learning task, evaluation should be done on a separate dataset.

### 3.6.1. FWI with MLP in extended domain



(a) FWI for Gaussian function.



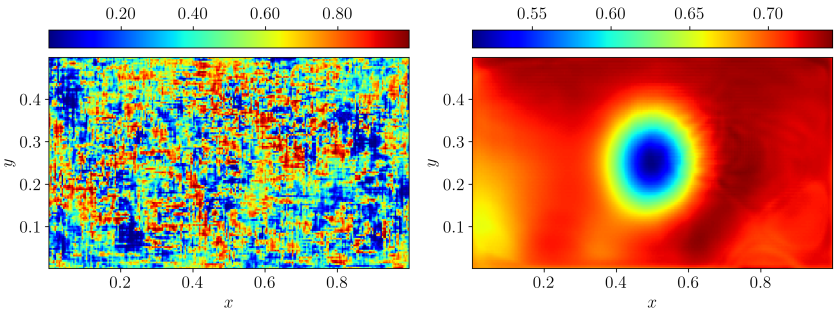
(b) FWI sine function.

Figure 3.30.: Plots show results for the FWI task for (a) Sigmoid of Gaussian function and (b) Sigmoid of sine function using MLP with 4 hidden layers and 350 neurons per layer. Trained for 1500 epochs with Xavier-Glorot initialization. The images on the left show the result after the first forward pass of the network. The images on the right show the converged result.

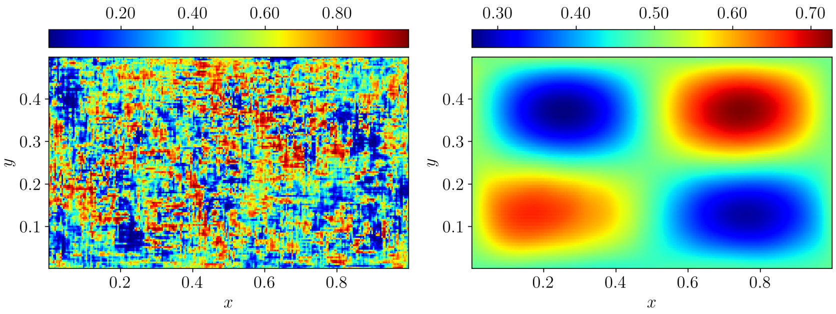
Observing the promising results obtained in the supervised learning setting displayed in figure 3.29 we attempt to solve the FWI task for the larger domain and both smooth functions. Our architecture performs well when approximating these functions in the defined domain, hence we can adventure ourselves to study the FWI problem. Figure 3.30 shows the results obtained after 1500 training epochs. The sigmoid of sine function is approximated with high accuracy, however the sigmoid of the inverted Gaussian function did not yield good results. When looking at the ground truth functions in the larger domain, we can see that the range of values in the images of the functions is around two times larger when looking at the sinusoidal shaped function. These larger range of values will make the propagated wave travel at different speeds inside the domain, leading to larger variations in the recorded measurements. This could be the reason behind our network being able to solve the FWI problem for sinusoidal material distribution and failing for the Gaussian shaped density distribution.

### 3.6.2. FWI with CNNs in extended domain

Although it was predictable, the CNN architecture can solve the FWI problem for the smooth functions represented in the larger domain. We included the results for completion in figures 3.31 and 3.32. Moreover, we also wanted to explore the difference in the results when initializing our architecture differently, in particular with Xavier-Glorot initialization, and with weights obtained from fitting a domain of ones in a supervised learning framework.

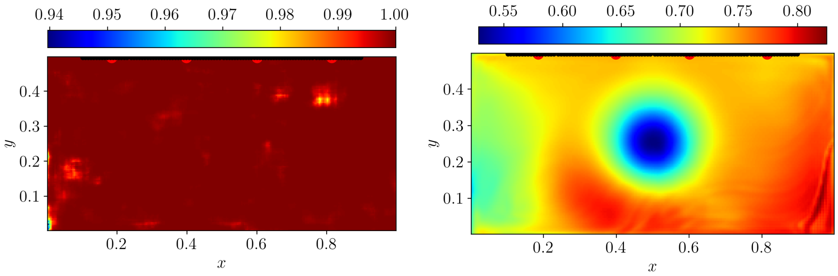


(a) FWI for Gaussian function.

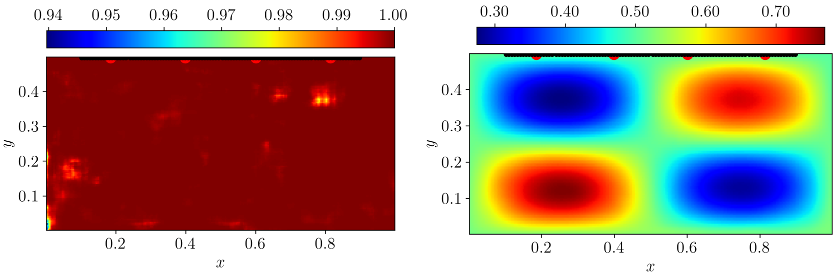


(b) FWI for sine function.

Figure 3.31.: Plots show results for the FWI task for (a) sigmoid of Gaussian function and (b) sigmoid of sine function using CNN with. Trained for 1500 epochs with coefficients initialized with Xavier-Glorot. The images on the left show the result after the first forward pass of the network. The images on the right show the converged result.



(a) FWI for Gaussian function.



(b) FWI for sine function.

Figure 3.32.: Plots show results for the FWI task for (a) sigmoid of Gaussian function and (b) sigmoid of sine function using CNN with. Trained for 1500 epochs with coefficients from supervised learning task of fitting a void-less domain. The images on the left show the result after the first forward pass of the network. The images on the right show the converged result.

We designed this experiment to see if initializing the weights and biases of a network to predict a domain of ones could be problematic or not, following the results obtained in 3.6. In this case, we did not use the SWIM network to fit a domain of ones, but we approximated a domain of ones in a supervised learning setting. The left images in figure 3.32 show this approximation was not perfect, but it was very close. The supervised learning task was trained for 5000 epochs. Comparing figures 3.32 and 3.31 seems that one initialization was better for one function, and the other initialization was better for the other. When approximating the Gaussian shaped function, it seems the Xavier-Glorot initialization was better and yielded better results when solving the FWI task. Contrary to this, it looks like initializing the network with the coefficients after fitting a domain of ones was better when solving the FWI problem with for the sinusoidal-shaped material distribution.

In addition, we do not include figures to quantify the error. The error when solving FWI problems does not seem as relevant as the actual shape of the obtained material distribution following the optimization procedure. The goal of FWI in the context of NDT is to capture the heterogeneities in the specimen we study. Even if the density is not approximated exactly, we can see the shape of the material distribution.

# 4. Conclusion

In this study, we have explored the potential of different neural network architectures to solve a full waveform inversion problem. Originally, the main objective of this work was to propose a new hybrid method, inspired by the work in [20], that would exploit the SWIM weight sampling algorithm to sample the weights of the hidden layer in a feed-forward neural network. We have carried out different experiments, described and explained in section 3.3. The first experiment was to solve the FWI problem in the most general case. This means we exploited a one-hidden layer feed-forward neural network and sampled the hidden layer weights using the SWIM algorithm. In addition, we fixed the hidden layer weights so that these stay the same during the optimization procedure and only train the linear output layer coefficients as well as the steepness and shift coefficient of the sigmoid function embedded after the output layer and described at the beginning of 3.4.1. Not knowing the ground truth material distribution, what is the best way to sample the weights using the SWIM algorithm? The specimen used in our FWI study is a rectangular domain with a small circular void, and therefore, most of the specimen is intact. This translates into having a specimen where the density value is one nearly everywhere, agreeing with the parameterization of voids described by equation 3.3. Hence, we use the SWIM network to fit a void-less domain, and once fitted, we use the internal weights of the SWIM network to initialize and freeze the hidden layer weights of our shallow feed-forward network. During training, we only optimized the output layer weights. The objective of this approach was to propose a new hybrid method to tackle the FWI task studied in [20]. Freezing the internal weights and biases would decrease the number of trainable parameters and would hopefully lead to faster inference times. Unfortunately, our experiments failed, and we could not solve the FWI problem with this approach - Using a shallow network, sampling the hidden layer's parameters with the SWIM algorithm, and optimizing the model's parameters using gradient descent for solving an FWI problem for a discontinuous material distribution.

Continuing with our investigation, we asked ourselves, how can we make it easier for our model to solve the FWI problem? Are there other weight initializations we can test to help our model converge to the true solution? The aim of experiments in section 3.4.2 was to initialize the hidden layer weights of our shallow network with the hidden layer weights of the SWIM network after fitting different domains. Each domain approximated by the SWIM network resembled closer and closer to the ground truth material distribution. Our aim was to make it progressively easier with each experiment for our network to retrieve the ground truth function in an

FWI setting. We have fitted a large circular void, an upwards shifted version of the true gamma function, and two right shifted versions of the true gamma function. Results for the big void initialization and upwards shifted ground truth initialization can be found in subsections 3.4.2. Appendix A contains the results of the other experiments, which have not been included in the main body of this work for being repetitive and not revealing extra details. Unfortunately, these experiments did not work either, and the converged result after training was not the ground truth material distribution. We can conclude that trying to solve the FWI problem for the discontinuous material distribution was not possible with our proposed architecture for the different initializations used and optimizing the model's parameters through gradient descent. Finally, as a last experiment for this section, we initialized both the hidden layer weights and bias terms as well as the output layer weights and bias term from our shallow network with all weights and bias terms from the SWIM network, and we only optimized the sigmoid parameters during the training loop. Each experiment performed was designed to make it easier for the model to yield the correct solution after training. After these experiments, we conclude that we were too optimistic to believe we could solve the FWI problem with our network architecture through gradient descent. The SWIM network is able to fit the ground truth gamma. Hence, there exists a set of weights and bias terms to yield the true material distribution. However, these are not achievable through gradient descent optimization methods and our simple architecture. As a last verification, we tried to solve the supervised learning problem for the ground truth function. We used the coordinates as inputs, representing equidistant grid points in our discretized domain, and the density value as the target function. We observed that our model cannot fit the target function precisely. The supervised learning problem is much simpler than the FWI problem. If this task fails, it is expected that the FWI task fails as well.

Striving to keep making the task simpler and simpler for our model, we turn to smooth functions, which are easier to approximate than functions containing jumps. After the experiments in section 3.4.1 we changed the material distribution to a smooth function. In particular, we conducted experiments for two smooth functions, described in subsection 3.5. We first keep the same domain dimensions and, hence, the same datasets for the FWI problem. Before attempting to solve the FWI problem with our network architecture, we conduct a supervised learning experiment to see if it is worth attempting to solve the, more challenging, FWI problem. On the one hand, MLPs yield disappointing results. Neither a feed-forward neural network with one hidden layer, nor a feed-forward neural network with three hidden layers, were able to solve the supervised learning task. Given the failure to complete the supervised learning task successfully, there is no point in attempting to solve the FWI problem. Training the MLP to solve the supervised task and FWI with gradient descent is challenging. On the other hand, the CNN architecture had no problem in solving both supervised learning tasks and FWI problems.

Finally, we realized that if we made the domain larger, MLPs could solve the supervised learning task. We made the domain ten times larger and adjusted some wave parameters in order to obtain a clean wave propagating in the new domain. The second row of table 3.2 shows the new parameters used in these experiments. Looking at the promising results obtained when solving the supervised learning task, we generated a new dataset to attempt solving the FWI problem with an MLP. The MLP used has four hidden layers and 500 neurons per layer. The results for the FWI task using this network are given in figure 3.30. The sinusoidal-shaped function is successfully retrieved with the MLP, and the FWI problem is solved quite accurately. However, the model does not perform well when solving the FWI task for the Gaussian-shaped material distribution. The first interesting observation to discuss is that the supervised task was solvable when considering a larger domain. This could be due to the variations of the function being less abrupt. When considering a larger domain, the rate of change of the function is lower than if we squeeze the function in a smaller region. We can see this as a less smooth function, causing problems for our simple model architecture. Extending the domain is another step forward in making the problem simpler for our model, simple enough to solve the supervised learning task with our simple architecture. Moreover, the second interesting observation to discuss is the fact that the FWI problem in the extended domain was only successful for the sinusoidal-shaped material distribution. My hypothesis for this observation is that the sinusoidal-shaped material distribution yields stronger variations in the wave received at the sensors compared to the Gaussian-shaped material distribution. We recall that when considering these smooth functions, the wavefields propagate in the entire domain and are only reflected at the boundaries. We can say that the Gaussian-shaped material distribution used is more similar to a void-less domain, and there is not a big difference with the values captured at the sensors when comparing them with the values at the sensors after propagating the wavefield in an intact material distribution.

After conducting our experiments we can draw several conclusions. Firstly, our proposed method to solve the FWI task by sampling the hidden layer parameters with the SWIM algorithm and only optimizing the output layer through gradient descent does not work to solve the FWI problem. More generally, we can say that training shallow networks with gradient descent is challenging, given the supervised learning experiment and FWI results. The only time we succeeded in training the shallow network was when we made the domain larger. In all experiments, we always input the whole domain into our network, and as such, we always used gradient descent to optimize our network rather than stochastic gradient descent, which can get stuck in local minima and cause problems in finding the optimal solution. By making the domain larger, it is possible to change the loss landscape and maybe start our optimization in a better position, less prone to getting stuck at a local minimum, or more likely to land at a better local minimum. In future work, it would be interesting to attempt the failed experiments using smaller batch sizes, to actually use stochastic gradient descent instead of gradient descent. It would be interesting to re-attempt making our method work, that is - using a one-hidden layer FNN, sampling the hidden layer parameters with the SWIM network, and optimizing the output layer, but this time making sure we use stochastic gradient descent by using minibatch training. In addition, we can also conclude that the most suitable initialization for solving the FWI problem depends on the function we want to recover, as shown by the experiments when solving the FWI problem with the CNN where we used two different initializations. In this work we only test two functions and two initializations for the CNN. In future work, we could test multiple ground truth functions in an FWI context, to see if either the Xavier-Glorot initialization, or the voidless domain initialization yields better results than the other when more material distribution functions are tested.

# Appendix

# A. Additional Experiments

## Further Initializations

### Initializing weights and bias terms to approximate a right-shifted version of the ground truth function

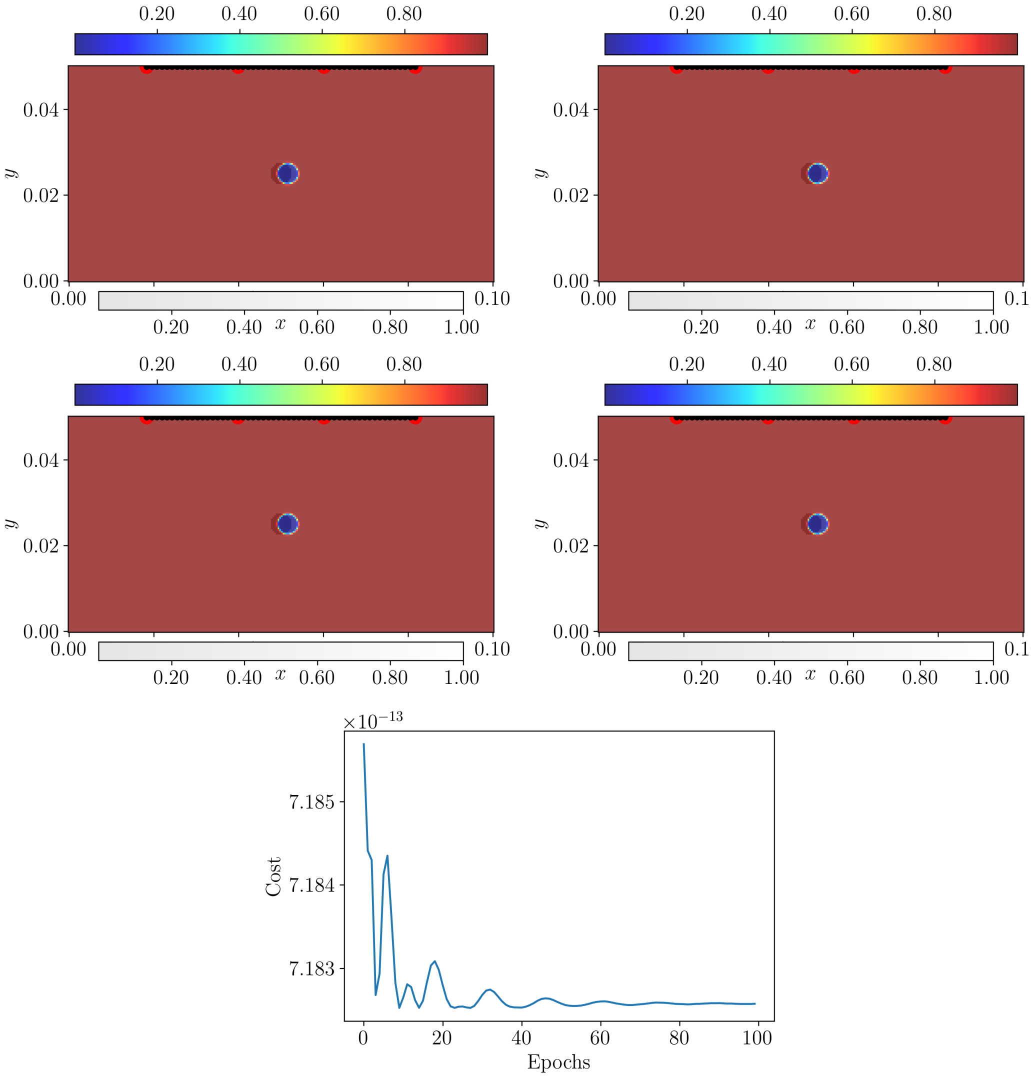


Figure A.1.: Plots show training results when initializing weights and bias terms with SWIM approximation of the true gamma shifted to the right. We show epochs 0, 25, 50 and 100.

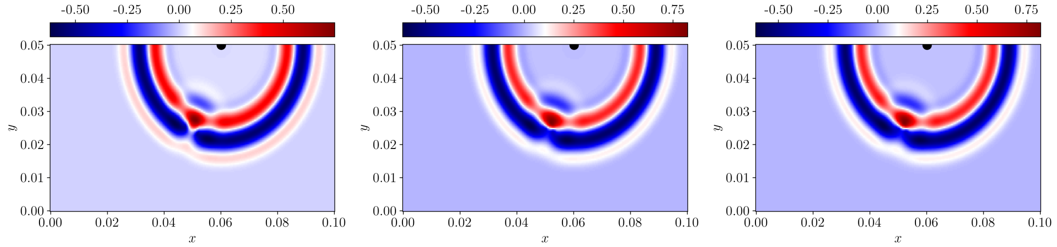


Figure A.2.: Plots show wavefield propagated in ground truth gamma, initialised gamma, and converged gamma

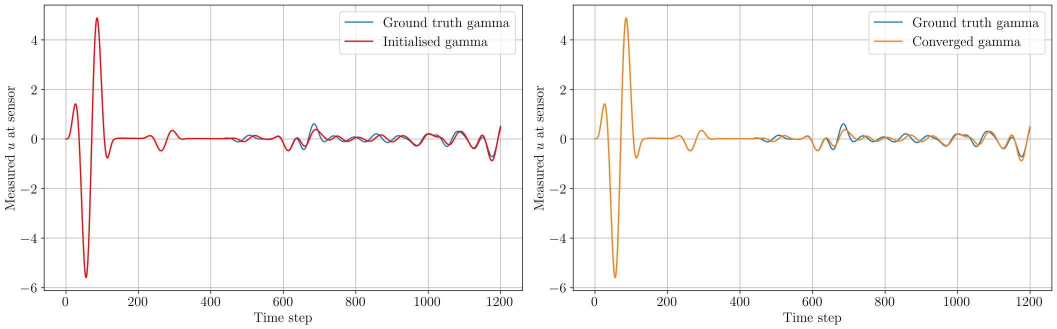


Figure A.3.: Wavefield propagated in ground truth gamma, initialised gamma, and converged gamma

### Initializing weights and bias terms to approximate a right-shifted version of the ground truth function and full domain information

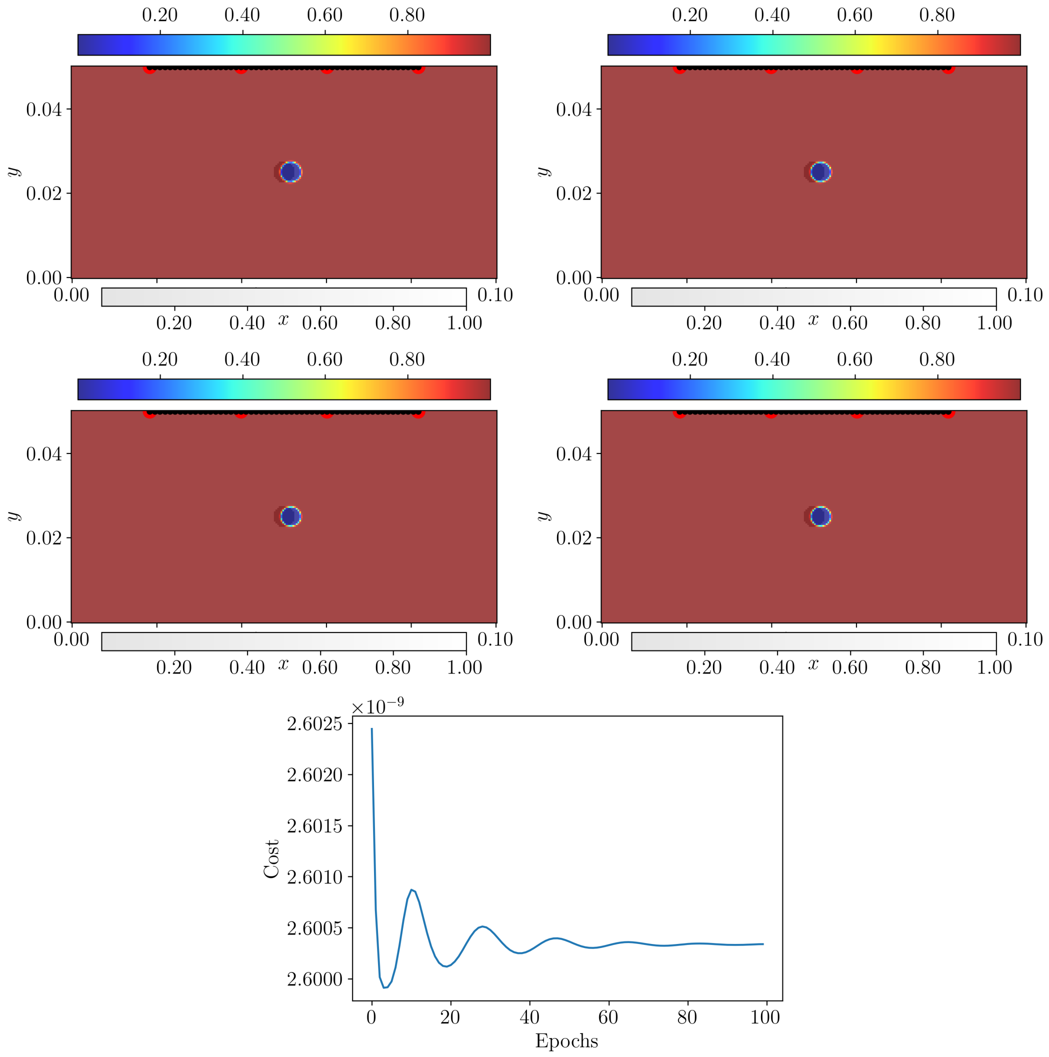


Figure A.4.: Plots show training results when initializing weights and bias terms with SWIM approximation of true gamma shifted to the right and full domain information. We show epochs 0, 25, 50 and 100.

### Initializing weights and bias terms to approximate a double right-shifted version of the ground truth function

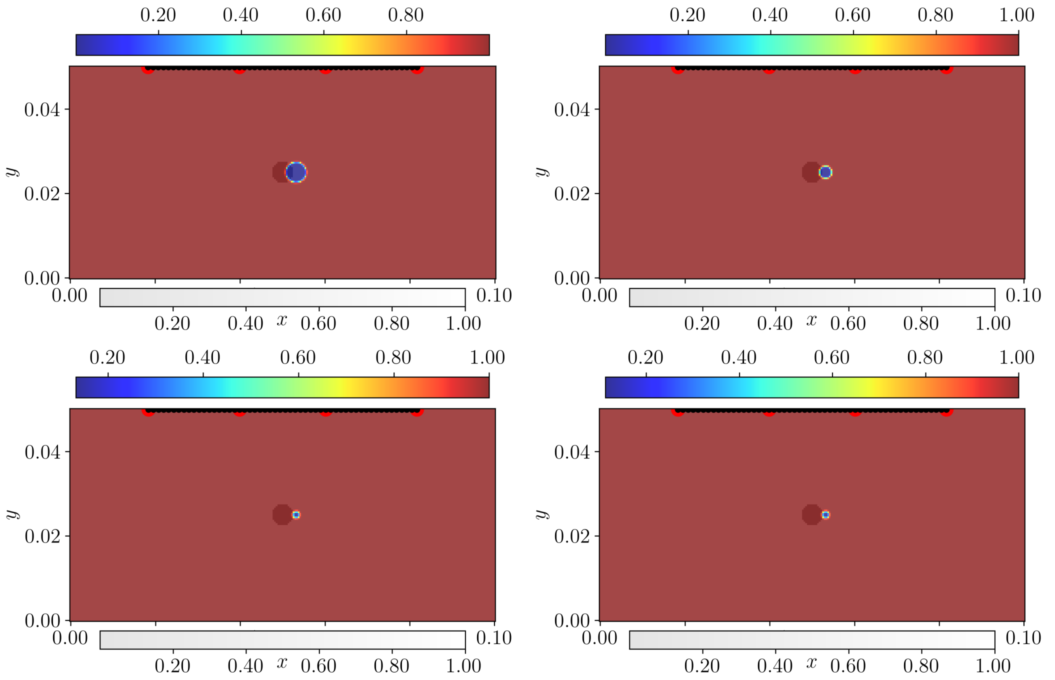


Figure A.5.: Plots show training results when initializing weights and bias terms with SWIM approximation of the ground truth gamma with a double right shift. We show epochs 0, 25, 50 and 100. Initial gamma is the true gamma shifted to the right by its diameter

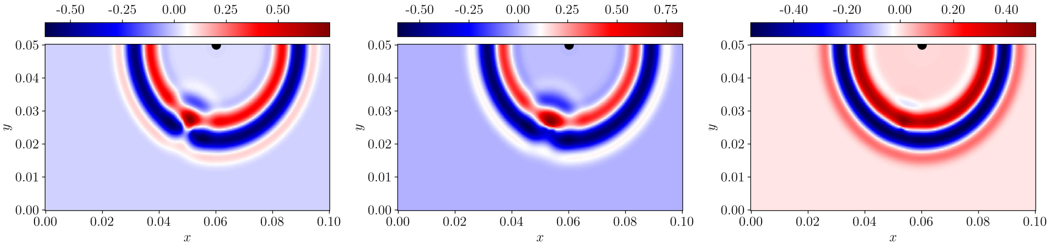


Figure A.6.: Plots show wavefields for timestep 220 for ground truth gamma, initialised gamma, and converged gamma.

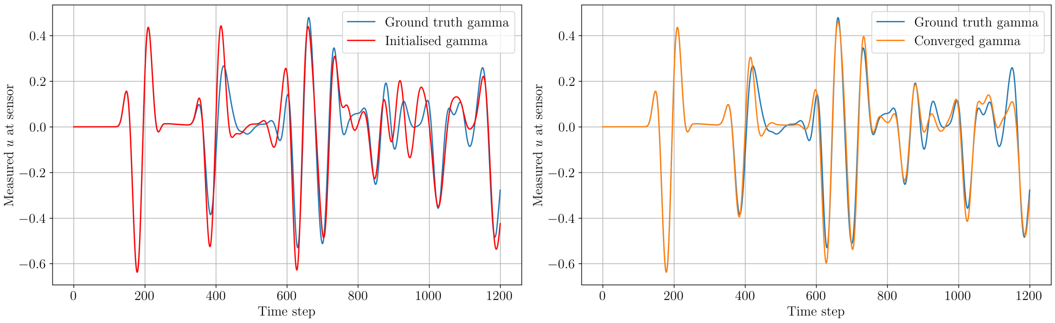


Figure A.7.: Plot shows time-series displacement measured at sensor at x position 152.

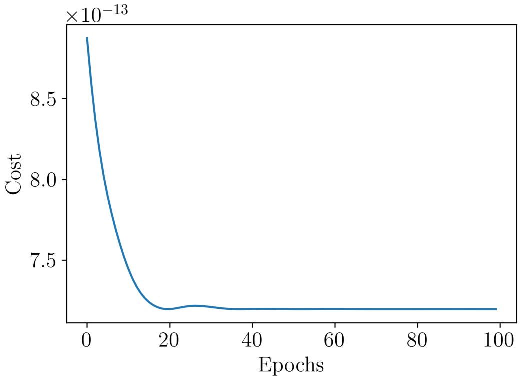


Figure A.8.: Plot shows cost evolution over 100 training epochs when initializing linear output layer coefficients to fit a double shifted version of the ground truth gamma.

# Bibliography

[1] Keiiti Aki, Anders Christoffersson, and Eystein S Husebye. Determination of the three-dimensional seismic structure of the lithosphere. *Journal of Geophysical Research*, 82:277–296, 1977.

[2] A. Bamberger, G. Chavent, Ch. Hemon, and P. Lailly. Inversion of normal incidence seismograms. *GEOPHYSICS*, 47:757–770, 1982.

[3] Atilim Gunes Baydin, Barak A. Pearlmutter, and Alexey Andreyevich Radul. Automatic differentiation in machine learning: a survey. *CoRR*, abs/1502.05767, 2015.

[4] Edip Baysal, Dan Kosloff, and John Sherwood. Reverse-time migration. *Geophysics*, 48:1514–1524, 11 1983.

[5] Erik Lien Bolager, Iryna Burak, Chinmay Datar, Qing Sun, and Felix Dietrich. Sampling weights of deep neural networks, 2023.

[6] Tim Bürchner, Philipp Kopp, Stefan Kollmannsberger, and Ernst Rank. Immersed boundary parametrizations for full waveform inversion. *Computer Methods in Applied Mechanics and Engineering*, 406:115893, March 2023.

[7] G. Chavent. Identification of function parameters in partial differential equations. In R.E. Goodson and Polis, editors, *Identification of parameter distributed systems*. ASME, New York, 1974.

[8] Jon F Claerbout. Toward a unified theory of reflector mapping. *GEOPHYSICS*, 36:467–481, 1971.

[9] Salvatore Cuomo, Vincenzo Schiano Di Cola, Fabio Giampaolo, Gianluigi Rozza, Maziar Raissi, and Francesco Piccialli. Scientific machine learning through physics-informed neural networks: Where we are and what's next. *Journal of Scientific Computing*, 92(88), 2022.

[10] Chitra Desai. Impact of weight initialization techniques on neural network efficiency and performance: A case study with mnist dataset. *International Journal Of Engineering And Computer Science*, 13(04), 2024.

[11] M. W. M. G. Dissanayake and N. Phan-Thien. Neural-network-based approximations for solving partial differential equations. *Communications in Numerical Methods in Engineering*, 10(3):195–201, 1994.

[12] Michael Dumbser and Martin Käser. An arbitrary high-order discontinuous galerkin method for elastic waves on unstructured meshes — ii. the three-dimensional isotropic case. *Geophysical Journal International*, 167(1):319–336, October 2006.

[13] Andreas Fichtner. *Full Seismic Waveform Modelling and Inversion*. Advances in Geophysical and Environmental Mechanics and Mathematics. Springer Science & Business Media, illustrated edition, 2010.

[14] Andreas Fichtner. *Full Seismic Waveform Modelling and Inversion*. Advances in Geophysical and Environmental Mechanics and Mathematics. Springer, Heidelberg, 2011.

[15] Wolfgang Friederich. The s-velocity structure of the east asian mantle from inversion of shear and surface waveforms. *Geophysical Journal International*, 153(1):88–102, 2003.

[16] Hongsun Fu et al. Seismic waveform inversion using a neural network-based forward. *Journal of Physics: Conference Series*, 1324:012043, 2019.

[17] Dan Givoli. A tutorial on the adjoint method for inverse problems. *Computer Methods in Applied Mechanics and Engineering*, 380:113810, 07 2021.

[18] L. Guasch, O. Calderón Agudo, and MX. et al. Tang. Full-waveform inversion imaging of the human brain. *npj Digit. Med.*, 3:28, 2020. Download citation.

[19] B. Gutenberg. Über die konstitution des erdinnern, erschlossen aus erdbebenbeobachtungen. *Physikalische Zeitschrift*, 14:1217–1218, 1913.

[20] Leon Herrmann, Tim Bürchner, Felix Dietrich, and Stefan Kollmannsberger. On the ultrasonice of neural networks for full waveform inversion. *Computer Methods in Applied Mechanics and Engineering*, 415:116278, October 2023.

[21] Leon Herrmann, Tim Bürchner, Felix Dietrich, and Stefan Kollmannsberger. On the use of neural networks for full waveform inversion [software], 2023.

[22] M. Hinze, editor. *Optimization with PDE Constraints*. Number 23 in Mathematical Modelling. Springer, Dordrecht, 2009.

[23] Kurt Hornik, Maxwell Stinchcombe, and Halbert White. Multilayer feedforward networks are universal approximators. *Neural Networks*, 2(5):359–366, 1989.

[24] Kurt Hornik, Maxwell Stinchcombe, and Halbert White. Multilayer feedforward networks are universal approximators. *Neural Networks*, 2(5):359–366, 1989.

[25] Ameya Jagtap, Kenji Kawaguchi, and George Karniadakis. Adaptive activation functions accelerate convergence in deep and physics-informed neural networks. *Journal of Computational Physics*, 404:109136, 11 2019.

[26] M Jannane, W Beydoun, E Crase, D Cao, Z Koren, E Landa, M Mendes, A Pica, M Noble, G Roeth, S Singh, R Snieder, A Tarantola, D Trezeguet, and M Xie. Wavelengths of earth structures that can be resolved from seismic reflection data. *GEOPHYSICS*, 54:906–910, 1989.

[27] J.B.W. Seismological tables. *Nature*, 151:472–472, 1916.

[28] Ting Jia. Advanced analysis of complex seismic waveforms to characterize the subsurface earth structure, 2011. Ph.D. dissertation, Columbia University.

[29] Wei jia Su and Adam M. Dziewonski. On the scale of mantle heterogeneity. *Physics of the Earth and Planetary Interiors*, 74(1):29–54, 1992. Special Issue Physical Properties of the Earth's Interior: Mantle to Core.

[30] Ehsan Kharazmi, Zongren Zhang, and George Em Karniadakis. hp-vpinns: Variational physics-informed neural networks with domain decomposition. *Comput. Methods Appl. Mech. Engrg.*, 374:113547, 2021.

[31] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization, 2017.

[32] S. Kollmannsberger, D. D'Angella, M. Jokeit, and L. Herrmann. *Physics-Informed Neural Networks*, volume 977. Springer International Publishing, Cham, 2021.

[33] S. Kollmannsberger, D. D'Angella, M. Jokeit, and L. Herrmann. *Physics-Informed Neural Networks*, volume 977. Springer International Publishing, Cham, 2021.

[34] Dimitri Komatitsch and Jean-Pierre Vilotte. The spectral element method: an efficient tool to simulate the seismic response of 2d and 3d geological structures. *Bulletin of the Seismological Society of America*, 88:368–392, 04 1998.

[35] Risi Kondor and Shubhendu Trivedi. On the generalization of equivariance and convolution in neural networks to the action of compact groups. In Jennifer Dy and Andreas Krause, editors, *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 2747–2755. PMLR, 10–15 Jul 2018.

[36] I.E. Lagaris, A. Likas, and D.I. Fotiadis. Artificial neural networks for solving ordinary and partial differential equations. *IEEE Transactions on Neural Networks*, 9(5):987–1000, 1998.

[37] I.E. Lagaris, A.C. Likas, and D.G. Papageorgiou. Neural-network methods for boundary value problems with irregular boundaries. *IEEE Transactions on Neural Networks*, 11(5):1041–1049, 2000.

[38] P. Lailly. The seismic inverse problem as a sequence of before stack migrations. In J.B. Bednar, E. Robinson, and A. Weglein, editors, *Conference on Inverse Scattering—Theory and Application*, pages 206–220, Philadelphia, 1983. SIAM.

[39] Gilles Lambaré, Jean Virieux, Raul Madariaga, and Side Jin. Iterative asymptotic inversion in the acoustic approximation. *GEOPHYSICS*, 57:1138–1154, 1992.

[40] Baodong Liu, Li Zeng, and Dongjiang Ji. Algebraic reconstruction technique class for linear scan ct of long object. 01 2008.

[41] Stéphane Mallat. Understanding deep convolutional networks. *Philosophical Transactions of the Royal Society A*, 374(2065), 2016.

[42] Stefano Markidis. The old and the new: Can physics-informed deep-learning replace traditional linear solvers? *Frontiers in Big Data*, 4, 2021.

[43] N.G.H. Meyendorf, P.B. Nagy, S.I. Rokhlin, R. Hull, R.M. Osgood, J. Parisi, and H. Warlimont, editors. *Nondestructive Materials Characterization*, volume 67 of *Springer Series in Materials Science*. Springer, Berlin, Heidelberg, 2004.

[44] John Michell. Conjectures concerning the cause, and observations upon the phaenomena of earthquakes; particularly of that great earthquake of the first of november, 1755, which proved so fatal to the city of lisbon, and whose effects were felt as far as africa, and more or less throughout almost all europe. *Philosophical Transactions of the Royal Society of London*, 51:566–634, 1760.

[45] Ben Moseley, Tarje Nissen-Meyer, and Andrew Markham. Deep learning for fast simulation of seismic waves in complex media. *Solid Earth*, 11:1527–1549, 08 2020.

[46] Masayasu Ohtsu. *Innovative AE and NDT Techniques for On-Site Measurement of Concrete and Masonry Structures: State-of-the-Art Report of the RILEM Technical Committee 239-MCM*. 01 2016.

[47] R. D. Oldham. Report on the great earthquake of 12th june, 1897. *Memoirs of the Geological Survey of India*, 29:379, 1899.

[48] Richard Dixon Oldham. The constitution of the interior of the earth, as revealed by earthquakes. *Quarterly Journal of the Geological Society of London*, 62(1-4):456–475, 1906.

[49] Houman Owhadi. Bayesian numerical homogenization. *Multiscale Modeling & Simulation*, 13(3):812–828, 2015.

[50] Dimitris C. Psichogios and Lyle H. Ungar. A hybrid neural network-first principles approach to process modeling. *AIChE Journal*, 38(10), 1992.

[51] Zhi Qu, Ping Jiang, and Wei Zhang. Development and application of infrared thermography non-destructive testing techniques. *Sensors*, 20(14):3851, 2020.

[52] M. Raissi, P. Perdikaris, and G. Karniadakis. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *Journal of Computational Physics*, 378:686–707, February 2019.

[53] Maziar Raissi and George Em Karniadakis. Hidden physics models: Machine learning of nonlinear partial differential equations. *Journal of Computational Physics*, 357:125–141, 2018.

[54] Maziar Raissi, Paris Perdikaris, and George Em Karniadakis. Inferring solutions of differential equations using noisy multi-fidelity data. *Journal of Computational Physics*, 335:736–746, 2017.

[55] Maziar Raissi, Paris Perdikaris, and George Em Karniadakis. Machine learning of linear differential equations ultrasonicing gaussian processes. *Journal of Computational Physics*, 348:683–693, 2017.

[56] J. Rao, M. Ratassepp, and Z. Fan. Guided wave tomography based on full-waveform inversion. *IEEE Trans Ultrason Ferroelectr Freq Control*, 63(5):737–745, May 2016. Epub 2016 Feb 29.

[57] Majid Rasht-Behesht, Christian Huber, Khemraj Shukla, and George Em Karniadakis. Physics-informed neural networks (pinns) for wave propagation and full waveform inversions. *Journal of Geophysical Research: Solid Earth*, 127(5), April 2022.

[58] C. Ravaut, S. Operto, L. Improta, J. Virieux, A. Herrero, and P. Dell'Aversana. Multiscale imaging of complex structures from multifold wide-aperture seismic data by frequency-domain full-waveform tomography: application to a thrust belt. *Geophysical Journal International*, 159(3):1032–1056, dec 2004.

[59] Daniel Rossato, Felipe Gutierrez, Giovanni Guarneri, Thiago Passarin, Gustavo Pires, and Daniel Pipa. Full waveform inversion for ndt using ultrasonic linear arrays. *Research and Review Journal of Nondestructive Testing*, 1, 08 2023.

[60] Samuel H. Rudy, Steven L. Brunton, Joshua L. Proctor, and J. Nathan Kutz. Data-driven discovery of partial differential equations, 2016.

[61] Robert Seidl. *Full Waveform Inversion for Ultrasonic Nondestructive Testing*. Dissertation, Ingenieurfakultät Bau Geo Umwelt, 2018.

[62] Robert Seidl and Ernst Rank. Full waveform inversion for ultrasonic flaw identification. *AIP Conference Proceedings*, 1806:090013, 2017.

[63] Khemraj Shukla, Patricio Clark Di Leoni, James Blackshire, Daniel Sparkman, and George Em Karniadakis. Physics-informed neural network for ultrasound nondestructive quantification of surface breaking cracks, 2020.

[64] Ali Siahkoohi, Mathias Louboutin, and Felix J. Herrmann. Neural network augmented wave-equation simulation, 2019.

[65] R Snieder, MY Xie, A Pica, and A Tarantola. Retrieving both the impedance contrast and background velocity: A global strategy for the seismic reflection problem. *GEOPHYSICS*, 54:991–1000, 1989.

[66] Jesper Spetzler, Jeannot Trampert, and Roel Snieder. Are we exceeding the limits of the great circle approximation in global surface wave tomography? *GEOPHYSICAL RESEARCH LETTERS*, 28(12):2341–2344, June 2001.

[67] A. Taffe, H. Wiggenhauser, and M. Raupach. Validierung zerstörungsfreier prüfverfahren im bauwesen. *Beton- und Stahlbetonbau*, 103(12):828–836, 2008.

[68] Carl Tape, Q. Y. Liu, Alessia Maggi, and Jeroen Tromp. Adjoint tomography of the southern california crust. *Science*, 325:988–992, 2009.

[69] Albert Tarantola. Inversion of seismic reflection data in the acoustic approximation. *GEOPHYSICS*, 49:1259–1266, 1984.

[70] Alexandre M. Tartakovsky, Cristina O. Marrero, Paris Perdikaris, et al. Physics-informed deep neural networks for learning parameters and constitutive relationships in subsurface flow problems. *Water Resources Research*, 56(5):e2019WR026731, 2020.

[71] F. A. C. Viana, R. G. Nascimento, A. Dourado, et al. Estimating model inadequacy in ordinary differential equations with physics-informed neural networks. *Comput. Structures*, 245:106458, 2021.

[72] Jean Virieux. P-sv wave propagation in heterogeneous media: Velocity-stress finite-difference method. *Geophysics*, 51:889–901, 01 1984.

[73] Jean Virieux and Stéphane Operto. An overview of full-waveform inversion in exploration geophysics. *Geophysics*, 74:WCC1–WCC26, 11 2009.

[74] Usman bin Waheed, Ehsan Haghighat, Tariq Alkhalifah, et al. Pinneik: Eikonal solution using physics-informed neural networks. *Comput. Geosci.*, 155:104833, 2021.

[75] Rachmasari Wardhani. Radiographic examination procedure as non destructive testing method in process piping. 2:1–8, 10 2019.

[76] Paul R. Williamson and M. H. Worthington. Resolution limits in ray tomography due to wave behavior: Numerical experiments. *GEOPHYSICS*, 58:727–735, 1993.

[77] John H. Woodhouse and Adam M. Dziewonski. Mapping the upper mantle: Three-dimensional modeling of earth structure by inversion of seismic waveforms. *Journal of Geophysical Research: Solid Earth*, 89(B7):5953–5986, July 1984.

[78] L. Yang, D. Zhang, and G. E. Karniadakis. Physics-informed generative adversarial networks for stochastic differential equations. *SIAM J. Sci. Comput.*, 42(1):A292–A317, 2020.

[79] K. Yomogida. Fresnel zone inversion for lateral heterogeneities in the earth. *PAGEOPH*, 138:391–406, 1992.

[80] Andreea Manuela Zelenyak, Reinhold Oster, Michael Mosch, Peter Jahnke, and Markus Sause. *Numerical modeling of ultrasonic inspection in fiber reinforced materials with explicit microstructure*. 01 2016.

[81] Zhongping Zhang and Youzuo Lin. Data-driven seismic waveform inversion: A study on the robustness and generalization, 2019.